CloneCademy

Qualitätssicherungsdokument

Gruppe 12:	Ilhan Simsiki <ilhan.simsiki@stud.tu-darmstadt.de> Leonhard Wiedmann <leonhard.wiedmann@stud.tu-darmstadt.de> Tobias Huber <tobias.huber@stud.tu-darmstadt.de> Claas Völcker <c.voelcker@stud.tu-darmstadt.de>
Teamleiter:	Alexander Nagl <alexander.nagl@t-online.de>
Auftraggeber:	iGEM-Team TU Darmstadt vertreten durch Thea Lotz <lotz@bio.tu-darmstadt.de> Fachbereich Biologie
Abgabedatum:	14.07.2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Bachelor-Praktikum SoSe 2017
Fachbereich Informatik

Contents

1. Einleitung **4**

2. Qualitätsziele **5**
2.1. Datensicherheit (Security) 5
2.2. Bedienbarkeit 6
2.3. Veränderbarkeit 7

A. Konkrete Ausführungen des QS-Maßnahmenkatalogs **10**
A.1. Ablaufplan der QS-Maßnahmen 10
A.1.1. Wöchentlich 10
A.1.2. Kalender 10
A.2. Checklisten 11
A.2.1. Checkliste Bedienbarkeit 11
A.2.2. Checkliste Datensicherheit 11
A.2.3. Checkliste Veränderbarkeit 12
A.3. Informelle Beschreibung der Nutzungsrollen 13
A.3.1. Nutzer*in 13
A.3.2. Moderator*in 13
A.3.3. Administrator*in 13
A.4. Rahmen der Wikidokumentation 13

B. Einleitung - Bericht und Anmerkung zum QS Prozess **14**
B.1. Struktur dieses Dokumentes 14
B.1.1. Nachweis über den QS Prozess pro Iteration 14
B.1.2. Nachweis über den QS Prozess pro Release 14
B.1.3. Weitere Belege 14
B.2. Änderungen des Prozesses während des Semesters 15
B.3. Durchführung der Maßnahmen kurz vor Abgabe 15

C. QS-Prozess nach Iteration und Release **16**
C.1. Iteration 7 - 29.06. 16
C.2. Iteration 8 - 06.07. 91
C.3. Iteration 9 - 13.07. 106
C.4. Iteration 10 - 20.07. 133
C.5. Iteration 11 - 27.07. 156
C.6. Iteration 12 - 03.08. 171
C.7. Iteration 13 - 10.08. 189
C.8. Iteration 14 - 17.08. 208
C.9. Iteration 15 - 24.08. 218
C.10. Iteration 16 - 31.08. 227
C.11. Iteration 17 - 07.09. 249

C.12.Iteration 18 - 14.09. 274
C.13.Iteration 19 - 21.09. 310
C.14.Release 1 - 07.08.2017 333
C.15.Release 2 - 13.09.2017 341
C.16.Release 3 - 01.10.2017 350

D. Progression des Design 353

E. Dokumentation des Codes - Auszüge 356

F. Testklassen und Coverage des finalen Produkts 371
F.1. Python Testklasse 371
F.2. Protractor/Selenium Testklasse 387
F.3. Coverage 389

G. GitHub-Wiki 427

H. Userstories 447

I. Protokolle der Auftraggeber*innentreffen 477

J. Endgültiger Stand des Projektes und Übergabe 495
J.1. Stand des Projektes bei Abgabe 495
J.2. Vereinbarungen zur Übergabe des Projektes 495

K. Abschließende Bemerkungen des Teams zum Projekt 496

Anmerkungen und Hinweise

Die Verfasser dieses Dokumentes verwenden als Maßnahme für die Gleichstellung aller Personen die sogenannte gendergerechte Sprache. Als Zeichen der Inklusion aller Geschlechter wird der Stern (*) oder eine Verlaufsform (*Nutzung* statt *Nutzer*) verwendet. Von dieser Regelung wird nur abgewichen, wenn entweder alle Mitglieder einer Gruppe sich ein und demselben Geschlecht zugehörig fühlen (z.B. im Entwicklerteam) oder die Vorgaben der Veranstalter*innen keine Alternative zulassen (z.B. auf dem Deckblatt). Die geschlechtergerechte Sprache wird auch von der Zentralen Frauenbeauftragten der TU Darmstadt empfohlen[1]

[1] https://www.tu-darmstadt.de/media/frauenbeauftragte/relaunch/pdf_10/zentrale_dokumente/Geschlechtergerecht_formulieren_digitale_Version.pdf

1 Einleitung

CloneCademy ist ein Projekt für das iGEM-Team der TU Darmstadt. Die *international Genetically Engineered Machine competition* (iGEM) ist ein internationaler Wettbewerb für Studierende auf dem Gebiet der synthetischen Biologie. Dieser wird seit 2003 von der iGEM-Foundation veranstaltet.

Im Rahmen des Wettbewerbs wird eine Online-Lernplattform für das iGEM-Team der TU Darmstadt erstellt. Das Ziel dieser Plattform ist es, durch interaktive Unterrichtseinheiten Prinzipien der Molekularbiologie und der synthetischen Biologie zu erlernen und sowohl eigene Lernfortschritte, als auch die anderer Teams begutachten zu können. Darüber hinaus soll es auch anderen Interessierten (z.B. andere iGEM-Teams, Lehrende an Universitäten und Schulen, etc.) möglich sein, eigene Inhalte einzupflegen und zur Verfügung zu stellen.

Als Kernfunktionalität bietet die Plattform Nutzer*innen die Möglichkeit, sich zu registrieren und einen Account anzulegen. Registrierte Nutzer*innen können bereitgestellte Aufgaben bearbeiten und Feedback zu ihren Antworten erhalten. Zusätzlich können Nutzer*innen mit Moderationsrechten neue Aufgaben in das System einpflegen und bereits bestehende überarbeiten. Die bereitgestellten Aufgabentypen sind Multiple-Choice-, Drag-And-Drop- und Lückentextaufgaben. Außerdem wird im Rahmen des vereinbarten Ziels der *Veränderbarkeit* die Möglichkeit geschaffen, nach Ende des Projektes auch noch weiter Aufgabentypen in die Plattform einzubauen.

Um einen Überblick über ihren bisherigen Lernerfolg zu bekommen, haben Nutzer*innen die Möglichkeit, Statistiken zu ihrer Nutzung der Plattform einzusehen. Den Auftraggeber*innen geben diese Statistiken auch Einblick in die Nutzung und die Qualität ihrer Inhalte, sodass diese weiter verbessert werden können.

Das Ziel des Projektes ist eine voll funktionsfähige Webanwendung für die aktuellen Versionen der Browser *Firefox* und *Chrome* auf Desktop- und Laptoprechner bereitzustellen. Eine eventuelle Erweiterung der Plattform, um zum Beispiel auch mobile Endgeräte zu unterstützen, wird durch die Implementierung einer REST-Schnittstelle ermöglicht.

2 Qualitätsziele

2.1 Datensicherheit (Security)

Im Rahmen des Projekts CloneCademy wird eine Webanwendung entwickelt, auf welche über das Internet zugegriffen werden kann. Daher ist die Sicherung gegen unbefugten Zugriff und unautorisierte Änderung der Daten in diesem Projekt ein wichtiges Qualitätsziel. Da CloneCademy sowohl persönliche Daten der Nutzer*innen als auch Metadaten über die Nutzung der Plattform und die Inhalte der einzelnen Lerneinheiten persistent speichert, ist es essentiell, dass Internetnutzer*innen diese Daten nicht verändern oder einsehen können, solange sie dazu nicht die benötigten Rechte besitzen.

Die größte Bedrohung geht von bekannten Webangriffen und Fehlkonfigurationen im Backend einer Webseite aus[1]. Während dieses Projekts wird die Plattform daher gegen die wichtigsten Sicherheitslücken einer Webanwendung gesichert. Dies sind vor allem mögliche Schwachstellen in der Nutzungsoberfläche und der REST-Schnittstelle.

Die betrachteten Angriffsvektoren sind:

- Möglichkeiten zur Ausführung fremden Codes (Injection & Cross-Site Scripting)
- Fehlerhafte Zugriffskontrolle
- Nutzung von Komponenten mit bekannten Schwachstellen

Um die Sicherheit der Anwendung zu gewährleisten, wird ein mehrschrittiger Plan befolgt.

Ein Entwickler des Teams wurde zum Sicherheitsbeauftragten ernannt, der die korrekte Durchführung aller genannten Maßnahmen sicherstellt. Seine Aufgabe ist es, die Einhaltung aller Programmierrichtlinien durchzusetzen und Tests zur Validierung der Schutzziele durchzuführen. Die genaue Vorgehensweise wird im Folgenden detailliert ausgeführt.

Um klare Zugriffsbeschränkungen umsetzen zu können, wurden Nutzungsrollen für das Projekt definiert (Administrator*in, Moderator*in und Nutzer*in)[2]. In jeder User Story wird festgehalten, ob und welche Zugriffsbeschränkungen einzelne Module der Webseite oder Daten besitzen. Um die Verwaltung der Rechte einzelner Nutzungsrollen zu ermöglichen, werden die zur Verfügung gestellten Authentifizierungswerkzeuge der genutzten Frameworks (Django, Django REST) verwendet. Die korrekte Implementierung der Rechteverwaltung wird im Rahmen der Security Reviews überprüft.

[1] Unsere Einschätzung relevanter Sicherheitslücken basiert auf den Empfehlungen des Open Web Application Security Project (OWASP)
Link zum Download: https://github.com/OWASP/Top10/raw/master/2017/OWASP%20Top%2010%20-%202017%20RC1-English.pdf

[2] Detaillierte Beschreibungen der Rollen finden sich im Anhang

Während der Implementierung halten alle Entwickler die Best Practices im Web Development ein. Als Referenz dafür werden die Handreichungen des Open Web Application Security Project (OWASP)[3] ausgeführt. In den wöchentlichen internen Code-Reviews wird die Qualität des Codes hinsichtlich dieser Richtlinien überprüft. Hierfür wird eine Checkliste[4] geführt, anhand der die Einhaltung dieser Richtlinien überprüft wird.

Um die Qualität des Codes im Python-basierten Backend automatisiert zu testen, wird das verbreitete Werkzeug *bandit*[5] verwendet. Als statisches Analyse-Werkzeug für das Frontend wird *ng2lint*[6] verwendet. Beide Programme unterstützen die Code-Review, indem die gelieferten Meldungen als Grundlage für eine detaillierte Analyse des Codes genutzt werden. Um eine dauerhaft hohe Qualität des Codes zu gewährleisten, werden die Analyse-Tools automatisiert bei einem Push auf dem Development-Branch des verwendeten Versionskontrollsystems Git verwendet. Alle gefundenen Schwachstellen werden vor der Präsentation der User Story zur Abnahme behoben. Dies ist die Aufgabe der entsprechenden Verantwortlichen für die User Story. Sollte dies nicht möglich sein, wird eine schriftliche Begründung verfasst und eine realistische Einschätzung über die Risiken der gefundenen Fehler an die Auftraggeber*innen gemeldet. Diese entscheiden, ob die Risiken tragbar sind und die User Story trotz der Sicherheitslücke als erfolgreich gemeldet wird.

Um die tatsächliche Sicherheit des Endproduktes gegen externe Angriffe zu zeigen, wird ein automatisiertes Penetrations-Werkzeug verwendet. Das OWASP empfiehlt hierfür das *Zed Attack Proxy Project*[7], welches eine Reihe bekannter Angriffe auf die Plattform ausführt und die gefundenen Schwachstellen meldet. Der Sicherheitsbeauftragte führt diese Tests spätestens eine Woche vor einem geplanten Release durch und meldet die Ergebnisse an die jeweiligen Entwickler des betroffenen Moduls, damit diese vor dem Release behoben werden können. Ist eine Behebung nicht möglich, werden die sicherheitsrelevanten Gefahren dieser Lücke zusammen mit möglichen weiteren Schritten zum Schutz der Anwendung im Betrieb an die Auftraggeber*innen gemeldet.

2.2 Bedienbarkeit

CloneCademy ist eine Lernplattform und steht als solche einer sehr heterogenen Gruppe an Nutzer*innen zur Verfügung. Daher muss die Plattform für alle Benutzer*innen einfach und intuitiv bedienbar sein. Das heißt, dass eine Bedienung ohne vorherige Einarbeitung möglich ist und komplexe Aufgaben mit einer Erklärung versehen sind. Um dies zu gewährleisten, wird das Design und der Aufbau aller Bereiche der Webseite nach diesem Kriterium evaluiert.

Auch für dieses Ziel wurde ein Mitglied des Teams benannt, um Nutzerstudien durchzuführen und die Einhaltung aller Richtlinien durchzusetzen.

[3] https://www.owasp.org/images/0/08/OWASP_SCP_Quick_Reference_Guide_v2.pdf
[4] Siehe Anhang
[5] https://wiki.openstack.org/wiki/Security/Projects/Bandit
[6] https://www.npmjs.com/package/ng2lint
[7] https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

Um das oben formulierte Ziel zu erreichen, verwenden wir als Grundlage für das Design der Seite das Material Design[8] für Angular2. Die Verwendung von standardisierten Komponenten sorgt dafür, dass die Webseite einen einheitlichen und übersichtlichen Aufbau hat. Um beispielsweise Knöpfe und andere Schaltflächen übersichtlicher zu gestalten, werden diese mit Icons[9] versehen. Bei der wöchentlichen internen Code-Review prüft der Beauftragte für dieses QS-Ziel anhand der Design-Checkliste[10] bei allen veränderten oder neuen Bereichen der Webseite, ob sie den Richtlinien eines einheitlichen Designs entsprechen. Sollte dies nicht der Fall sein, wird die User Story den Auftraggeber*innen nicht präsentiert, sondern in der nächsten Iteration überarbeitet und erneut geprüft.

Da die Bedienbarkeit der Oberfläche nicht objektiv gemessen werden kann, werden zusätzlich zu den Designrichtlinien Nutzerstudien durchgeführt. In den Studien setzen sich die Proband*innen ohne vorherige Erklärung mit der Plattform auseinander und werden darum gebeten, verschiedene Aufgaben zu erfüllen[11]. Während der Studie wird das Verhalten der Nutzer*innen mit Hilfe eines Screen-Capture-Programms erfasst, um später zu überprüfen, ob die Aufgaben zielgerichtet gelöst werden konnten. Zusätzlich wird die persönliche Meinung der Proband*innen nach der Studie mit einem Fragebogen[12] erfasst und ausgewertet.

Um abschließend ein, von einer Vielzahl unterschiedlicher Nutzer*innen gut bedienbares, Produkt zu haben, werden die Nutzerstudien in mehreren Iterationen und mit unterschiedlichen Gruppen durchgeführt. Verbesserungsvorschläge aus den vorherigen Studien können so direkt wieder getestet werden. Die Nutzerstudien erfolgen mindestens eine Woche vor einem geplanten Release oder wenn die Auftraggeber*innen eine Zwischenevaluation des Designs wünschen.

Nach jeder Nutzerstudie wird ein Maßnahmenkatalog mit den Problemen und Verbesserungsvorschlägen der Proband*innen erstellt, welche ein dafür benannter Entwickler bis zum nächsten Treffen mit den Auftraggeber*innen umsetzt. Diese können dann die Designänderungen überprüfen und erst durch die Abnahme der Designänderungen ist die Iteration der Studie abgeschlossen.

2.3 Veränderbarkeit

Für die Webanwendung CloneCademy ist es wichtig, dass sie nach Abschluss des Projekts noch veränderbar ist. Es muss für weitere Entwicklerteams möglich sein, sowohl neue Inhalte in die Datenbank einzupflegen, als auch den Quellcode der Webseite selbst erweitern und verändern zu können, da die Auftraggeber*innen planen, die Webseite nach dem Abschluss des Bachelorprojektes eigenständig weiterzuentwickeln. Um diese Veränderbarkeit zu gewährleisten, wird

[8] https://material.angular.io/
[9] https://material.io/icons/
[10] siehe Anhang
[11] Ablauf siehe Anhang
[12] siehe Anhang

im Projekt auf folgende Punkte geachtet:

- hohe Qualität des Quellcode
- ausführliche Kommentare
- externe Dokumentation
- ausführliche Testabdeckung des Codes

Es wurde ein Verantwortlicher für das Qualitätsziel benannt, der Ansprechpartner für alle Rücksprachen oder Fragen ist und dafür sorgt, dass die Qualität der oben aufgeführten Bereiche, wie unten beschrieben, sichergestellt wird.

Qualität des Quellcodes Grundlegend werden während der Entwicklung der Software die Styleguides der verwendeten Programmiersprachen und Frameworks umgesetzt. Diese sind der *Angular Style Guide*[13] für das Frontend und *PEP 8*[14] für das Backend. Um die Einhaltung dieser beiden Richtlinien zu überprüfen, werden bei jedem Push auf dem Development-Branch statische Analyse-Tools verwendet, welche überprüfen, ob alle Richtlinien umgesetzt wurden.

Die verwendeten Tools sind *pep8*[15], welches die Einhaltung der generellen Python Richtlinien überprüft, *django-lint*[16], welches Framework spezifische Fehler markiert (z.B. Verwendung von veralteten Methoden) und *ng2lint*[17], welches den Frontend-Code überprüft. Alle Fehler, die von diesen Tools gefunden wurden, werden bis zum nächsten internen Gruppentreffen von demjenigen Entwickler behoben, der für die User Story verantwortlich war, bevor der Code den Auftraggeber*innen zur Abnahme präsentiert wird.

Kommentare Jede Methode und jede Klasse im Code erhält einen Kommentar. Zusätzlich werden unklare Stellen im Code (z.B. komplexe Steuermechanismen oder Abfragen) mit eigenen Erklärungen versehen. Alle Kommentare werden in Englisch verfasst.

Bei der Code-Review überprüft der Beauftragte anhand der Veränderbarkeits-Checkliste[18], ob alle Klassen und Methoden kommentiert sind und ob die Kommentare das Verstehen des Codes unterstützen. Den Auftraggeber*innen werden nur User Stories mit vollständig dokumentiertem Code zur Abnahme präsentiert.

Wiki Das Projekt verwendet zur klaren Trennung zwischen Back- und Frontend eine sogenannte REST-API. Diese stellt eine Schnittstelle dar, über welche die Daten des Backends abgerufen und gegebenenfalls verändert oder erweitert werden können. Diese wird deshalb gesondert dokumentiert, da eine korrekte Implementierung und Nutzung Grundlage für die Interaktion zwischen Front- und Backend ist.

[13] https://angular.io/guide/styleguide
[14] https://www.python.org/dev/peps/pep-0008/
[15] https://pypi.python.org/pypi/pep8
[16] https://pypi.python.org/pypi/django-lint
[17] https://www.npmjs.com/package/ng2lint
[18] siehe Anhang

Um die Schnittstelle zentral zu dokumentieren, wird ein Wiki geführt, in dem alle Funktionen der Schnittstelle zwischen Backend und Frontend definiert sind. Ist für eine User Story eine neue Funktion nötig, wird diese von allen Entwicklern zusammen entworfen und dann im Wiki dokumentiert.

Um sicherzustellen, dass die Dokumentation der Schnittstelle vollständig ist, überprüft der Beauftragte spätestens eine Woche vor Release, ob alle Funktionen aufgeführt sind und ob die Dokumentation den vorgegebenen Rahmen[19] erfüllt. Bevor das Wiki nicht vollständig ist, wird die Software nicht zum Release freigegeben.

Testabdeckung Bei Codeerweiterung muss bereits bestehender Code weiterhin funktionieren. Um dies zu verifizieren, wird eine umfangreiche Testabdeckung des Codes sichergestellt. Das heißt, dass eine vollständige Function Coverage und eine Statement Coverage von mindestens 80% erreicht wird. Eine Überprüfung der Abdeckung erfolgt im Backend durch das mitgelieferte Testframework des Django-Projekts. Im Frontend werden Tests mit Hilfe des Frameworks Selenium[20] durchgeführt.

Die Function Coverage wird vor jeder internen Code-Review sichergestellt. Die Tests der API Schnittstellen werden bei deren Definition geschrieben, damit sie den jeweiligen Entwicklern als Vorlage für eine korrekte Implementierung dienen können. Da eine umfassende Statement Coverage während des Projektes nicht zu jeder Zeit implementiert werden kann, ist diese zu jedem Release anzufertigen. Damit wird gezeigt, dass der Code stabil ist und die Tests können als Basis für die Implementierung der kommenden User Stories verwendet werden.

Eine User Story wird erst zur Abnahme präsentiert, wenn für alle Funktionen Tests vorliegen. Ein Release wird erst freigegeben, wenn eine Code Coverage von mindestens 80% erreicht ist. Die Überprüfung dieser Regeln erfolgt durch den, für das QS-Ziel zuständigen Entwickler.

[19] siehe Anhang
[20] http://www.seleniumhq.org/

A Konkrete Ausführungen des QS-Maßnahmenkatalogs

A.1 Ablaufplan der QS-Maßnahmen

A.1.1 Wöchentlich

Das Entwicklerteam trifft sich wöchentlich, um den aktuellen Stand des Projektes zu besprechen. Für die Qualitätssicherung werden dabei folgende Aufgaben durchgeführt:

- Sichten aller Reports der automatisierten Tools
- Überprüfen des aktuellen Designs
- Überprüfen der Checklisten
- Beschluss, welche User Stories zur Abnahme präsentiert werden

Alle hierbei gefundenen Mängel werden von den jeweiligen, für den Code im Rahmen einer User Story verantwortlichen Entwickler behoben. Eine User Story wird erst präsentiert, wenn der Code keine Mängel mehr aufweist.

A.1.2 Kalender

- Anfang August: geplanter erster Release auf dem Server des iGEM-Teams
- Anfang September: geplanter zweiter Release auf dem Server des iGEM-Teams
- Ende September: endgültiger Release der finalen Software auf dem Server des iGEM-Teams

Eine Woche vor jedem Release:

- Deadline der Nutzerstudie
- Security Test mit dem *ZED Attack Tool*
- Prüfung der Wiki auf Vollständigkeit

Alle hierbei gefundenen Mängel werden bis zum jeweiligen Release behoben. Zur Behebung wird ein dedizierter Entwickler benannt.

A.2 Checklisten

A.2.1 Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- □ Wurden nur Elemente des Material Designs verwendet?
- □ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- □ Sind Buttons durch ihre Position direkt ersichtlich?
- □ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- □ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- □ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?
- □ Ist alles in der Mindestauflösung (1024x768) erkennbar?

A.2.2 Checkliste Datensicherheit

Bei der Codereview wird von folgendem Bedrohungsmodell ausgegangen:

- Angreifende: Es wird davon ausgegangen, dass böswillige Nutzer*innen der Plattform bei der Beantwortung der Fragen unerlaubte Methoden benutzen möchten, sowie dass Dritte die gespeicherte Nutzerdaten auslesen und verändern wollen.
- Angriffsoberfläche: Als Oberfläche wird die API und das Frontend betrachtet.
- Mögliche Angriffe: Es werden die gebräuchlichsten Angriffsformen wie zum Beispiel XSS, CSRF sowie Code- und SQL-Injection erwartet.
- Als grundlegende Sicherheitsmaßnahmen wird die korrekte Implementierung von Django, Django REST und Angular2 verwendet.

Fragen zu jedem Codeabschnitt:

- □ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- □ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- □ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- □ Werden alle Nutzereingaben nach Fehlern gefiltert?

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

A.2.3 Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?
- ☐ Gibt es obsoleten Code?
- ☐ Gibt es unbenutzte Variablen?
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?
- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:
  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:
  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert
- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

Tests

- ☐ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

A.3 Informelle Beschreibung der Nutzungsrollen

Die konkreten Rechte jeder Nutzungsrolle ergeben sich aus den Beschreibungen der User Stories. Rollen sind hierarchisch strukturiert. übergeordnete Rollen erhalten auch immer alle Rechte der untergeordneten Rollen. Ein*e Administrator*in hat zum Beispiel alle Rechte, die auch normale Nutzer*innen und die Moderator*innen besitzen.

A.3.1 Nutzer*in

Nutzer*innen besitzen ein Account, mit welchem sie sich gegenüber der Webseite authentifizieren können. Sie können Aufgaben lösen und Feedback erhalten. Außerdem haben sie Zugriff auf ihre eigene Nutzerseite, können die eigenen Daten ändern und ihre Statistik einsehen.

A.3.2 Moderator*in

Moderator*innen können neue Lerninhalte auf der Webseite hochladen. Sie sind nur dazu berechtigt, Kurse zu verändern, die sie selbst angelegt haben. Bevor ein angelegter Kurs für alle Nutzer*innen sichtbar wird, muss er durch einen Administrator überprüft werden.

Einzelnen Moderator*innen können das Recht zur Freigabe von Kursen erhalten. Diese können dann ihre eigenen Kurse sofort, ohne erneute Überprüfung, freischalten.

A.3.3 Administrator*in

Administrator*innen haben das Recht, anderen Nutzer*innen zusätzliche Rechte (Moderation, Administration) zu geben und zu entziehen.

A.4 Rahmen der Wikidokumentation

Für jede Funktion der Schnittstelle muss das Wiki folgende Informationen enthalten:

- Name der Schnittstelle
- Beschreibung der Funktionsweise
- erlaubte Methoden (POST/GET/DELETE)
- erwartete Header-Felder
- erwartete Parameter aus der URL
- gültige Formatierung des JSON-Objekts (bei POST-Methoden)
- gültige Formatierung des JSON-Objekts in der Antwort
- Übersicht aller möglichen Antwortcodes und ihrer Bedeutung

B Einleitung - Bericht und Anmerkung zum QS Prozess

B.1 Struktur dieses Dokumentes

Im nachfolgenden Teil des Dokumentes befinden sich die Nachweise über die durchgeführten Maßnahmen. Wir haben uns beste Mühe gegeben, dies übersichtlich zu gestalten. Aufgrund der vielen auomatisierten Outputs und der Fülle an Checklisten haben wir uns entschlossen, den Anhang in mehrere Teile zu trennen. Im ersten Teil haben wir alle Checklisten und die wöchentlichen automatisierten Tests angehängt. Darauf folgend haben wir einen Übersicht über die Designveränderung während des Prozesses und die Dokumentation der Code Kommentare, des Testoutputs und unseres Entwicklungs-Wikis angehängt.

Alle Dateien, vor allem die Checklisten und die Userstories, sowie die Quelldateien der Tests und Codebespiele finden sich im beiligenden Ordner, damit diese auch gesondert eingesehen werden können. Dort finden sich auch die Videoaufzeichnungen der Nutzerstudien.

B.1.1 Nachweis über den QS Prozess pro Iteration

Um dieses Dokument nicht unübersichtlich werden zu lassen, wurden nicht alle Outputs der automatisierten Tests angehängt. Wir haben stattdessen die Meldungen zum Stand der wöchentlichen Abgabe gespeichert, um den Verlauf der Softwareentwicklung deutlich zu machen. Zusätzlich sind alle Checklisten für die jeweils abgegebenen Userstories angehängt. Wurden noch Anmekrungen gemacht, so ist direkt auf den Checklisten vermerkt wurden, wann diese verbessert wurden.

Um die Testabdeckung zu dokumentieren haben wir zu jeder Iteration die Übersichtsseite der Reports angehängt, und einen gesamten Report für das finale Produkt.

B.1.2 Nachweis über den QS Prozess pro Release

Zu jedem Release finden sich die Protokolle der Nutzerstudie und die ausgefüllten Fragebögen. Zusätzlich haben wir die Reports des ZAP-Tools angehängt und eine kurze Stellungsnahme zu den erfolgten Maßnahmen.

B.1.3 Weitere Belege

Um die Dokumentation der Webseite im Wiki anschaulich zu dokumentieren, haben wir das gesamte Wiki bei Stand der Abgabe angehängt.

Um die Dokumentation zum QS-Ziel "Bedienbarkeit" anschaulich zu gestalten, haben wir die Veränderungen an der Webseite durch Screenshots dokumentiert. Zusätzlich finden sich die Checklisten (pro Iteration) und die aus den Nutzungsstudien entstandenen Dokumente (Protokolle und Issues in GitHub) im folgenden Anhang.

B.2 Änderungen des Prozesses während des Semesters

Während des Semesters ist dem Team aufgefallen, dass der vorgestellte QS-Prozess nicht optimal ist. Die wichtigste Änderung ist, dass die automatisiere Code-Überprüfung im Backend durch das Tool *pylint* mit dem *pylint_django* Plugin durchgeführt wurde. Dieses vereinigt die Funktionen von *pep8* und *django-lint*, ist einfacher konfigurier- und verwendbar. Das Team hat deshalb beschlossen, dieses Tool zu verwenden, obwohl im QS-Dokument ein anderes vorgeschlagen wurde. Des Weiteren haben wir beschlossen, im Python Backend eine Code Qualität von 9.5/10 Punkten anzustreben, da eine Behebung aller Fehler durch Eigenheiten der Tools nicht immer möglich war.

B.3 Durchführung der Maßnahmen kurz vor Abgabe

Einige QS-Maßnahmen wurden noch einmal kurz vor Abgabe des Projektes an die Auftraggeber*innen durchgeführt, auch wenn die Zeit an diesem Punkt nicht mehr ausreicht, um alle gefundenen Probleme noch zu beheben. Dies hat den Hintergrund, dass die Auftraggeber*innen explizit wünschen, dass das Produkt auch in Zukunft noch weiter entwickelt und verfeinert wird. Somit haben wir beschlossen, einen letzten Durchlauf der Nutzungsstudien zu machen, um somit dem iGEM-Team die Möglichkeiten dazu an die Hand zu geben.

C QS-Prozess nach Iteration und Release

C.1 Iteration 7 - 29.06.

Wir haben unseren QS-Prozess das erste Mal zur 7. Iteration durchgeführt, allerdings noch nicht vollständig (einige Details fehlen noch, z.B. geänderte Tools). Die ersten Auszüge sind entstanden, in dem wir alle Tests und Tools einmal durchgeführt haben, ohne etwas an der Software zu verändern. Dadurch haben wir einen Gesamtüberblick gewonnen. Die gefundenen Fehler wurden in den kommenden Iterationen inkrementell behoben, wenn eine Datei für eine Userstory geändert werden musste.

Abgegebene Userstories (bislang): 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 14, 15, 16

Commit Hash der ersten Durchführung: 22f76fcce2d1d6b03857a9a4db7f9276433aecdd

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 1

Reviewer: Claas
Entwickler: Tobias
Commithash: Gesamtüberprüfung der bisherigen US
Datum: 30.06.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?

  Ja

- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?

  Ja

- ☐ Sind Buttons durch ihre Position direkt ersichtlich?

  Ja

- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?

  Ja

- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?

  Ja

- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

  Ja

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar? ja

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  ja

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

  ja

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

  ja

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  ja

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  ja (keine aufgetreten)

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  ja (ziemlich viele, wird in den kommenden Iterationen behoben)

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  nein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  Ja

- ☐ Für das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?

    Ja

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    Ja

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      Doku steht noch nicht, muss noch diskutiert werden

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

    nein (wird noch verbessert) / ja

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      ~~Ja~~ n

Kommentare

- ☐ Sind alle Klassen kommentiert?

  nein (wurde verbessert)

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ✓

  - ☐ Autor

    ✓

- ☐ Sind alle Methoden kommentiert?

  nein (wurde direkt verbessert)

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ✓

  - ☐ Autor

    ✓

  - ☐ Eingabeparameter

    ✓

  - ☐ Rückgabewert

    ✓

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nein, Code ist verständlich

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollständig?

  Struktur noch nicht klar, siehe oben

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  nein (wird in den nächsten Iterationen nachgeseicht

- ☐ Wie hoch ist die Statement Coverage?

  54%

4

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US-02

Reviewer: ilhan

Entwickler: leo

Commithash:

Datum:

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- [x] Wurden nur Elemente des Material Designs verwendet?
- [x] Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art? Die Schaltflächen sind nicht alle in einer Reihe
- [x] Sind Buttons durch ihre Position direkt ersichtlich?
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- [x] Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben? Schlumb -> Login
- [x] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools? Ja, bessern!
- ☐ Gibt es obsoleten Code? Nein
- ☐ Gibt es unbenutzte Variablen? Nein
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? nein

- ☐ Wurden Hilfsmethoden ausgelagert? Nein
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert? Ja
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Kommentare fehlen im Frontend komplett!

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig? Bitte eintrag ins Wiki!

Tests

- ☐ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 3

Reviewer: Tobias
Entwickler: Claas
Commithash: Base
Datum: Gesamtüberprüfung

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- [x] Wurden nur Elemente des Material Designs verwendet?

icon fehlt
fixed

- [x] Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- [x] Sind Buttons durch ihre Position direkt ersichtlich?
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- [x] Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- [x] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)? Nein, eigenes Fix
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☑ Gibt es Meldungen der Tools? ja, viele, wird verbessert
- ☑ Gibt es obsoleten Code?
- ☑ Gibt es unbenutzte Variablen? ja, wird verbessert
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert?

- [x] Wurden Hilfsmethoden ausgelagert?

- [x] Für das Frontend:

  - [x] Sind alle POST Methoden richtig formatiert?

- [x] Für das Backend:

  - [x] Sind alle Schnittstellen wie abgesprochen implementiert?

    - [x] Wenn ja: Sind diese begründet und dokumentiert?

  - [x] Geben alle Views einen gültigen und passenden Status Code zurück?

  - [x] Wurde die Datenbankenstruktur geändert?

    - [x] Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- [x] Sind alle Klassen kommentiert? fehlt, wird verbessert

  Dies umfasst:

  - [x] Beschreibung der Funktion und Verwendung

  - [x] Autor

- [x] Sind alle Methoden kommentiert? fehlt, wird verbessert

  Dies umfasst:

  - [x] Beschreibung der Funktion und Verwendung
  - [x] Autor
  - [x] Eingabeparameter
  - [x] Rückgabewert

- [x] Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- [x] Sind die Kommentare verständlich für alle Entwickler?
- [x] Ist die Dokumentation im Wiki vollständig?

Tests

- [x] Sind Tests aller Methoden vorhanden?
- [ ] Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 5

Reviewer: Leonhard Wiedmann
Entwickler: Ilhan Simsiki
Commithash: base der evaluation
Datum: Gesamtüberprüfung aller Userstorys.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?
  ~~Es wurde noch kein Material design verwendet~~
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
  ja
- ☐ Sind Buttons durch ihre Position direkt ersichtlich?
  ja
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
  ja
- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
  ja
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?
  ja

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

ja

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

ja

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

ja

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

ja

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

ja

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

ja

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

ja viele. wird nachgebessert

- ☐ Gibt es obsoleten Code?

Nein

- ☐ Gibt es unbenutzte Variablen?

ja, wird nachgebessert

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

Nein

- ☐ Wurden Hilfsmethoden ausgelagert?
  Nein, keine Hilfsmethoden
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
    ~~Ja~~ keine neuen Post methoden
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    Ja
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
      Ja
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
    Ja
  - ☐ Wurde die Datenbankenstruktur geändert?
    Nein
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?
  Ja
  Dies umfasst:
  - ☐ Beschreibung der Funktion und Verwendung
    Ja
  - ☐ Autor
    Ja

3

- ☐ Sind alle Methoden kommentiert?

  ja

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ja

  - ☐ Autor

    ja

  - ☐ Eingabeparameter

    ja

  - ☐ Rückgabewert

    ja

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  Nein

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollständig?

  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  ja

- ☐ Wie hoch ist die Statement Coverage?

  Base der Evaluation

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US – 06

Reviewer: ilhan
Entwickler: Claas
Commithash: 6d187
Datum: 07.06

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- [x] Wurden nur Elemente des Material Designs verwendet? Schaltflächen waren nicht im Material design, wurde beim Treffen gefixed
- [x] Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- [x] Sind Buttons durch ihre Position direkt ersichtlich?
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- [x] Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- [x] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- [x] Ist alles in der Mindestauflösung (1024x768) erkennbar? wurde angepasst

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- [x] HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- [x] Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- [x] Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- [x] Werden alle Nutzereingaben nach Fehlern gefiltert?
- [x] Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  keine Meldungen der Tools

Checkliste Veränderbarkeit

Code Qualität

- [x] Gibt es Meldungen der Tools?
- [x] Gibt es obsoleten Code?
- [x] Gibt es unbenutzte Variablen?
- [x] Wurden bereits vorhandene Funktionalitäten neu implementiert?

- [x] Wurden Hilfsmethoden ausgelagert?
- [x] Für das Frontend:
  - [x] Sind alle POST Methoden richtig formatiert?
- [x] Für das Backend:
  - [x] Sind alle Schnittstellen wie abgesprochen implementiert?
    - [x] Wenn ja: Sind diese begründet und dokumentiert?
  - [x] Geben alle Views einen gültigen und passenden Status Code zurück?
  - [x] Wurde die Datenbankenstruktur geändert?
    - [x] Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- [x] Sind alle Klassen kommentiert?

  Dies umfasst: Statistics Component

  - [x] Beschreibung der Funktion und Verwendung
  - [x] Autor (wurden beim Treffen nachgetragen)

- [x] Sind alle Methoden kommentiert?

  Dies umfasst: downloadstatistics()

  - [x] Beschreibung der Funktion und Verwendung
  - [x] Autor
  - [x] Eingabeparameter
  - [x] Rückgabewert

- [x] Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

- [x] Sind die Kommentare verständlich für alle Entwickler?

- [ ] Ist die Dokumentation im Wiki vollständig?

  wird noch gemacht (14.06 erledigt)

Tests

- [x] Sind Tests aller Methoden vorhanden?

- [x] Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 7 (lt. 4)

Reviewer: Tobias

Entwickler: Leonhard

Commithash:

Datum: 8.06.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet? ~~Input anpassen~~ fixed 3.08.
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich? ~~fixe position würbesse~~
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)? ~~Nein~~ fixed 3.08.
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools? ja, viele
- ☑ Gibt es obsoleten Code?
- ☐ Gibt es unbenutzte Variablen? werden nochmal diskutiert manche sinnvoll
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert?

7

- ☑ Wurden Hilfsmethoden ausgelagert?
- ☑ Für das Frontend:
  - ☑ Sind alle POST Methoden richtig formatiert?
- ☑ Für das Backend:
  - ☑ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☑ Wenn ja: Sind diese begründet und dokumentiert?
  - ☑ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☑ Wurde die Datenbankenstruktur geändert?
    - ☑ Wenn ja: Sind diese begründet und dokumentiert? hinzufügen 3. OS.

Kommentare

- ☑ Sind alle Klassen kommentiert? Nein nachgeholt 3. OS.

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor

- [x] Sind alle Methoden kommentiert? ~~Nein~~ Nachgeholt 3.08.

  Dies umfasst:

  - [x] Beschreibung der Funktion und Verwendung
  - [x] Autor
  - [x] Eingabeparameter
  - [x] Rückgabewert

- [x] Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- [x] Sind die Kommentare verständlich für alle Entwickler?
- [x] Ist die Dokumentation im Wiki vollständig? ~~Nein~~ fixed 3.08.

Tests

- [x] Sind Tests aller Methoden vorhanden? ~~Nein~~ fixed
- [ ] Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US -08

Reviewer: Ilhan

Entwickler: Leon

Commithash: 6d187

Datum: 07.06

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?

  Name der Buttons in der Besprechung in sinnvollere Namen geändert
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

→ wurde angepasst

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☑ Gibt es Meldungen der Tools? Nein
- ☑ Gibt es obsoleten Code? Nein
- ☑ Gibt es unbenutzte Variablen? Nein
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert? Nein
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst: MultipleChoiceQuestion-Component

  Im Frontend fehlen ALLE Kommentare!!

  - 08.06 OK!
- ☐ Beschreibung der Funktion und Verwendung
- ☐ Autor

3

- ☐ Sind alle Methoden kommentiert? X

  Dies umfasst: submit() fehlt! – 08.06 ok!

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert
- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

  ok!

Tests

- ☐ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 3

Reviewer: Claus
Entwickler: Leon
Commithash: Gesamtüberprüfung bisheriger US
Datum: 02.07.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  keine Inputs vorhanden

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

  ja

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

  ja

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  ja

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  ja (keine aufgetreten)

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  ja (viele) siehe restliche US der Gesamtüberprüfung

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  ja → refactoring

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  ja → refactoring

- ☐ Wurden Hilfsmethoden ausgelagert?

  ja

- ☐ Für das Frontend:

  - ☑ Sind alle POST Methoden richtig formatiert?

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    ja

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

    /

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

    ja

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein (bzw. aktuell nicht wichtig, da Grundlage)

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  ja

  Dies umfasst:

  ja

  - ☐ Beschreibung der Funktion und Verwendung

    ja

  - ☐ Autor

    ja

3

- ☐ Sind alle Methoden kommentiert?

  nein (einige im Frontend fehlen, wird behoben)

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nein

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollständig?

  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  nein

- ☐ Wie hoch ist die Statement Coverage?

  54 %

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 10

Reviewer: Leonhard Wiedmann
Entwickler: Ilhan Simsiki
Commithash: base der evaluation
Datum:

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss f r die Elemente folgende Checkliste beachtet werden:

- [x] Wurden nur Elemente des Material Designs verwendet?
- [ ] Sind Schaltfl chen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
  Das Select Element ist ~~optisch~~ kaum erkennbar. Wird nachgebessert.
- [ ] Sind Buttons durch ihre Position direkt ersichtlich?
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- [x] Sind die Schaltfl chen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- [ ] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?
  Select Element

1

☒ Ist alles in der Mindestaufl sung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

☒ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

☒ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

☒ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

☒ Werden alle Nutzereingaben nach Fehlern gefiltert?

☒ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

☒ Gibt es Meldungen der Tools?

☐ Gibt es obsoleten Code?
Nein

☐ Gibt es unbenutzte Variablen?
Nein

☐ Wurden bereits vorhandene Funktionalit ten neu implementiert?
Nein

- ☐ Wurden Hilfsmethoden ausgelagert?
  Nein, keine Hilfsmethoden

- ☐ F r das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?
    keine Post Methoden.

- ☐ F r das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    ja

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?
      ja

  - ☐ Geben alle Views einen g ltigen und passenden Status Code zur ck?
    ja

  - ☐ Wurde die Datenbankenstruktur ge ndert?
    Nein

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?
  Nein wird nachgereicht

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
    ~~ja~~ Nein

  - ☐ Autor
    Nein

- ☐ Sind alle Methoden kommentiert?

  Nein, wird noch gearbeitet

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ✓

  - ☐ Autor

    ✓

  - ☐ Eingabeparameter

    ✓

  - ☐ R ckgabewert

    ✓

- ☐ Gibt es weitere, schwer verst ndliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  Nein

- ☐ Sind die Kommentare verst ndlich f r alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollst ndig?

  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  ja

- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 11

Reviewer: Tobias

Entwickler: Claas

Commithash: cf525f

Datum: 06.07.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- [x] Wurden nur Elemente des Material Designs verwendet?
- [x] Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- [x] Sind Buttons durch ihre Position direkt ersichtlich? mehr padding/margin
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- [x] Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- [x] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☑ Gibt es Meldungen der Tools? ja, einige ✓
- ☑ Gibt es obsoleten Code?
- ☑ Gibt es unbenutzte Variablen?
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert?

11

- [x] Wurden Hilfsmethoden ausgelagert?
- [x] Für das Frontend:
  - [x] Sind alle POST Methoden richtig formatiert?
- [x] Für das Backend:
  - [x] Sind alle Schnittstellen wie abgesprochen implementiert?
    - [x] Wenn ja: Sind diese begründet und dokumentiert?
  - [x] Geben alle Views einen gültigen und passenden Status Code zurück?
  - [x] Wurde die Datenbankenstruktur geändert?
    - [x] Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- [x] Sind alle Klassen kommentiert?

  Dies umfasst:

  - [x] Beschreibung der Funktion und Verwendung
  - [x] Autor

- [x] Sind alle Methoden kommentiert?

  Dies umfasst:

  - [x] Beschreibung der Funktion und Verwendung
  - [x] Autor
  - [x] Eingabeparameter
  - [x] Rückgabewert

- [x] Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- [x] Sind die Kommentare verständlich für alle Entwickler?
- [x] Ist die Dokumentation im Wiki vollständig?

Tests

- [x] Sind Tests aller Methoden vorhanden? ~~fehlen!~~
- [ ] Wie hoch ist die Statement Coverage?

74%

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US - 14

Reviewer: ilhan
Entwickler: Leon
Commithash: 22f76
Datum: 30.06

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?

  X – continue nicht im md!
  – 01.07 OK!

- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools? Nein
- ☐ Gibt es obsoleten Code? Nein
- ☐ Gibt es unbenutzte Variablen? Nein
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

- ☐ Wurden Hilfsmethoden ausgelagert? Keine
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert? Keine
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert? Nein
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☑ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten) Nein

- ☑ Sind die Kommentare verständlich für alle Entwickler?

- ☐ Ist die Dokumentation im Wiki vollständig? alles Vollständig!

  ↳ Keine

Tests

- ☑ Sind Tests aller Methoden vorhanden?

- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 15

Reviewer: Tobias
Entwickler: Ilhan
Commithash: Basis
Datum: ~~[illegible]~~ 29.06.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☑ Gibt es Meldungen der Tools? ja, einige fixed
- ☑ Gibt es obsoleten Code?
- ☑ Gibt es unbenutzte Variablen?
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert?

15

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

15

- ☑ Sind alle Methoden kommentiert?

  - Dies umfasst:

    - ☑ Beschreibung der Funktion und Verwendung
    - ☑ Autor
    - ☑ Eingabeparameter
    - ☑ Rückgabewert

- ☑ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☑ Sind die Kommentare verständlich für alle Entwickler?
- ☑ Ist die Dokumentation im Wiki vollständig?

Tests

- ☑ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

74 %

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 16

Reviewer: Chaos

Entwickler: Leon

Commithash: Gesamtüberprüfung bisheriger US

Datum: 02.07.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

es gibt keine Inputs → keine Security probleme

☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

n a

☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

n a

☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

keine gefunden

Checkliste Veränderbarkeit

Code Qualität

☐ Gibt es Meldungen der Tools?

ng lint " Fehler (wurde behoben)

☐ Gibt es obsoleten Code?

nein

☐ Gibt es unbenutzte Variablen?

nein

☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  na

- ☐ Für das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?

    Ja

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    Ja

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      ✓

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

    nein (Error nicht gefangen, wurde behoben)

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      ✓

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Ja

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung

  - ☑ Autor

- ☐ Sind alle Methoden kommentiert?

  Ja

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nein

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollständig?

  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  nein, wird in den nächsten Iteration behoben

- ☐ Wie hoch ist die Statement Coverage?

  5 %

Veränderbarkeit

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
npm info lifecycle clonecademy@0.0.0~lint: clonecademy@0.0.0
[..................] - : info lifecycle clonecademy@0.0.0~lint: clonecademy@0.0

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/error-message/error-message.component.ts[20, 3]:
   ↪ Implement lifecycle hook interface OnInit for method ngOnInit in class
   ↪ ErrorMessageComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/service/error.service.ts[12, 21]: missing whitespace
ERROR: /angular/src/app/service/error.service.ts[13, 11]: Identifier 'dialogRef
   ↪ ' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/loader/loader.component.ts[10, 42]: expected nospace
   ↪ before colon in parameter
ERROR: /angular/src/app/loader/loader.component.ts[12, 3]: Implement lifecycle
   ↪ hook interface AfterViewInit for method ngAfterViewInit in class
   ↪ LoaderComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/service/server.service.ts[84, 7]: comment must start
   ↪ with a space
ERROR: /angular/src/app/service/server.service.ts[35, 1]: Exceeds maximum line
   ↪ length of 140
ERROR: /angular/src/app/service/server.service.ts[33, 23]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[42, 50]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[44, 16]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[55, 20]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[61, 20]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[64, 18]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[74, 62]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[76, 16]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[90, 32]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[96, 32]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[99, 30]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[115, 47]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[143, 22]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[147, 20]: missing whitespace
```

```
ERROR: /angular/src/app/service/server.service.ts[153, 18]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[51, 9]: Identifier 'options'
   ↪ is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/server.service.ts[85, 9]: Identifier 'options'
   ↪ is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/server.service.ts[126, 15]: Identifier '
   ↪ response' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/server.service.ts[117, 9]: Identifier 'headers'
   ↪  is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/server.service.ts[119, 9]: Identifier 'body' is
   ↪  never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/server.service.ts[121, 9]: Identifier 'options'
   ↪  is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/server.service.ts[30, 34]: " should be '
ERROR: /angular/src/app/service/server.service.ts[35, 42]: " should be '
ERROR: /angular/src/app/service/server.service.ts[35, 67]: " should be '
ERROR: /angular/src/app/service/server.service.ts[123, 75]: " should be '
ERROR: /angular/src/app/service/server.service.ts[128, 27]: " should be '
ERROR: /angular/src/app/service/server.service.ts[129, 27]: " should be '
ERROR: /angular/src/app/service/server.service.ts[44, 7]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[55, 11]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[61, 11]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[64, 11]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[76, 7]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[90, 23]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[96, 23]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[99, 23]: missing whitespace
ERROR: /angular/src/app/service/server.service.ts[137, 98]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[13, 17]: Type boolean trivially
   ↪  inferred from a boolean literal, remove type annotation
ERROR: /angular/src/app/service/user.service.ts[18, 55]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[36, 20]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[50, 34]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[51, 34]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[52, 50]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[53, 40]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[61, 19]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[65, 23]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[69, 18]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[79, 19]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[82, 9]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[82, 5]: misplaced 'else'
ERROR: /angular/src/app/service/user.service.ts[5, 24]: " should be '
ERROR: /angular/src/app/service/user.service.ts[38, 61]: " should be '
ERROR: /angular/src/app/service/user.service.ts[66, 54]: " should be '
ERROR: /angular/src/app/service/user.service.ts[53, 32]: == should be ===
```

```
ERROR: /angular/src/app/service/user.service.ts[51, 7]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[52, 10]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[53, 11]: missing whitespace
ERROR: /angular/src/app/service/user.service.ts[79, 7]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[14, 23]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[16, 26]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[17, 48]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[18, 62]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[19, 48]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[28, 9]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[42, 16]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[54, 23]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[56, 28]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[71, 11]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[71, 7]: misplaced 'else'
ERROR: /angular/src/app/service/course.service.ts[59, 22]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[62, 17]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[62, 13]: misplaced 'else'
ERROR: /angular/src/app/service/course.service.ts[73, 18]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[76, 13]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[76, 9]: misplaced 'else'
ERROR: /angular/src/app/service/course.service.ts[87, 18]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[88, 46]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[89, 35]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[15, 9]: Identifier 'value' is
    ↪  never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/course.service.ts[58, 15]: Identifier 'value'
    ↪ is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/course.service.ts[72, 13]: Identifier 'value'
    ↪ is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/course.service.ts[29, 61]: " should be '
ERROR: /angular/src/app/service/course.service.ts[43, 61]: " should be '
ERROR: /angular/src/app/service/course.service.ts[19, 42]: == should be ===
ERROR: /angular/src/app/service/course.service.ts[89, 29]: == should be ===
ERROR: /angular/src/app/service/course.service.ts[16, 7]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[17, 10]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[18, 12]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[19, 13]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[56, 9]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[59, 15]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[73, 11]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[88, 8]: missing whitespace
ERROR: /angular/src/app/service/course.service.ts[89, 9]: missing whitespace
ERROR: /angular/src/app/app.component.ts[14, 78]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[38, 1]: Exceeds maximum line
    ↪ length of 140
```

```
ERROR: /angular/src/app/login/login.component.ts[42, 14]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[43, 19]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[50, 9]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[50, 5]: misplaced 'else'
ERROR: /angular/src/app/login/login.component.ts[56, 41]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[47, 13]: Identifier 'dialogRef
    ↪ ' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/login/login.component.ts[5, 29]: " should be '
ERROR: /angular/src/app/login/login.component.ts[8, 24]: " should be '
ERROR: /angular/src/app/login/login.component.ts[21, 14]: " should be '
ERROR: /angular/src/app/login/login.component.ts[22, 18]: " should be '
ERROR: /angular/src/app/login/login.component.ts[24, 14]: " should be '
ERROR: /angular/src/app/login/login.component.ts[25, 18]: " should be '
ERROR: /angular/src/app/login/login.component.ts[51, 29]: " should be '
ERROR: /angular/src/app/login/login.component.ts[38, 79]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[43, 7]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[56, 7]: missing whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[45, 9]: comment
    ↪ must start with a space
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[28, 32]: missing
    ↪  whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[42, 40]: missing
    ↪  whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[44, 20]: missing
    ↪  whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[43, 9]:
    ↪ Identifier 'percent' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[15, 30]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[15, 40]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[16, 30]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[16, 40]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[48, 12]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[44, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[2, 26]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[3, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[15, 34]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[16, 29]: " should be '
```

```
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/menu/menu.component.ts[16, 72]: missing whitespace
ERROR: /angular/src/app/menu/menu.component.ts[17, 34]: " should be '
ERROR: /angular/src/app/learn/course/course.component.ts[17, 14]: Type boolean
    ↪ trivially inferred from a boolean literal, remove type annotation
ERROR: /angular/src/app/learn/course/course.component.ts[47, 13]: Shadowed
    ↪ variable: 'data'
ERROR: /angular/src/app/learn/course/course.component.ts[32, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[55, 60]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[53, 13]: Identifier '
    ↪ lastModule' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course/course.component.ts[54, 13]: Identifier '
    ↪ lastQuestion' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course/course.component.ts[46, 44]: " should be '
ERROR: /angular/src/app/learn/course/course.component.ts[63, 29]: " should be '
ERROR: /angular/src/app/learn/course/course.component.ts[46, 33]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[46, 34]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[55, 11]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/module.directive.ts[10, 29]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[10, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[22, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[26, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[29, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[2, 31]: " should be '
ERROR: /angular/src/app/learn/question/question.module.ts[23, 12]: " should be
    ↪ '
ERROR: /angular/src/app/learn/question/question.module.ts[7, 13]: The selector
    ↪ of the component "QuestionModule" should be named kebab-case and include
    ↪  dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/question/question.module.ts[10, 14]: The name of
    ↪ the class QuestionModule should end with the suffix Component (https://
    ↪ goo.gl/5X1TE7)
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[33, 7]: comment must start with a space
```

```
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[26, 16]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[29, 20]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[28, 14]: Identifier 'ans' is never reassigned; use 'const'
    ↪  instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[27, 9]: Identifier 'sendAnswer' is never reassigned; use '
    ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[3, 32]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[4, 31]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 21]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 46]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 69]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 49]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 50]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 45]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 46]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 31]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 32]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[29, 9]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[7, 13]: The selector of the component "
    ↪ MultipleChoiceQuestionComponent" should be named kebab-case and include
    ↪ dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[16, 3]: Implement lifecycle hook interface OnInit for
    ↪ method ngOnInit in class MultipleChoiceQuestionComponent (https://goo.gl
    ↪ /w1Nwk3)
ERROR: /angular/src/app/learn/question/question.component.ts[1, 1]: Exceeds
    ↪ maximum line length of 140
ERROR: /angular/src/app/learn/question/question.component.ts[88, 13]: Shadowed
    ↪ variable: 'data'
```

```
ERROR: /angular/src/app/learn/question/question.component.ts[105, 58]: Shadowed
    ↪  variable: 'data'
ERROR: /angular/src/app/learn/question/question.component.ts[111, 79]: Shadowed
    ↪  variable: 'data'
ERROR: /angular/src/app/learn/question/question.component.ts[59, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[63, 17]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[84, 11]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[92, 22]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[97, 27]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[108, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[108, 5]: misplaced
    ↪  'else'
ERROR: /angular/src/app/learn/question/question.component.ts[101, 27]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[104, 11]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[104, 7]: misplaced
    ↪  'else'
ERROR: /angular/src/app/learn/question/question.component.ts[117, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[118, 27]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[121, 5]: misplaced
    ↪  'else'
ERROR: /angular/src/app/learn/question/question.component.ts[121, 30]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[125, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[125, 5]: misplaced
    ↪  'else'
ERROR: /angular/src/app/learn/question/question.component.ts[76, 11]:
    ↪ Identifier 'question' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/question/question.component.ts[86, 9]: Identifier
    ↪  'data' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/question/question.component.ts[3, 31]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[7, 49]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[9, 32]: " should
    ↪ be '
```

```
ERROR: /angular/src/app/learn/question/question.component.ts[64, 21]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[64, 46]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[64, 69]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[87, 22]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[87, 47]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[87, 70]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[105, 28]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[111, 26]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[127, 33]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[130, 31]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[130, 56]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[130, 80]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[64, 49]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 50]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 45]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 31]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 32]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[87, 50]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[87, 51]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[87, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[87, 47]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[87, 32]: missing
    ↪ whitespace
```

```
ERROR: /angular/src/app/learn/question/question.component.ts[87, 33]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[130, 59]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[130, 55]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[130, 56]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[130, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[130, 42]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[97, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[101, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[118, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[121, 12]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[40, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[9, 35]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[10, 49]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[20, 5]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 21]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 46]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 49]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 50]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 45]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 31]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 32]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[28, 14]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.module.ts
    ↪ [9, 21]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.module.ts
    ↪ [10, 12]: " should be '
```

```
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.module.ts
    ↪ [8, 14]: The name of the class AddQuestionModule should end with the
    ↪ suffix Component (https://goo.gl/5X1TE7)
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[1, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[25, 16]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[35, 11]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[39, 14]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[43, 26]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[30, 9]: Identifier 'question' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[40, 9]: Identifier 'response' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[36, 23]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[43, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[18, 14]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[26, 30]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[30, 14]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[3, 35]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[14, 57]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[20, 13]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[31, 30]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[31, 42]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [80, 7]: comment must start with a space
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [1, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [56, 16]: missing whitespace
```

```
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [57, 40]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [72, 20]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [78, 10]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [83, 16]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [84, 54]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [90, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [95, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [101, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [103, 55]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [106, 21]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [60, 11]: Identifier 'question' is never reassigned; use 'const' instead
    ↪  of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [61, 11]: Identifier 'q' is never reassigned; use 'const' instead of '
    ↪ let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [107, 13]: Identifier 'save' is never reassigned; use 'const' instead of
    ↪  'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [104, 11]: Identifier 'tmp' is never reassigned; use 'const' instead of
    ↪ 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [105, 11]: Identifier 'index' is never reassigned; use 'const' instead
    ↪ of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [102, 9]: Identifier 'values' is never reassigned; use 'const' instead
    ↪ of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [3, 38]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [8, 44]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [36, 12]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [84, 25]: " should be '
```

```
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [85, 23]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [91, 21]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [96, 21]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [57, 27]: != should be !==
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [84, 22]: == should be ===
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [84, 45]: == should be ===
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [57, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [84, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [103, 8]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [106, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [72, 3]: Implement lifecycle hook interface AfterViewInit for method
    ↪ ngAfterViewInit in class AddModuleComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[30, 5]: if statements must be braced
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[1, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[29, 13]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[49, 14]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[56, 27]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[60, 7]: misplaced 'else'
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[60, 28]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[65, 7]: misplaced 'else'
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[65, 30]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[74, 17]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[78, 9]: missing whitespace
```

```
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[81, 53]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[61, 13]: Identifier 'index' is never reassigned; use 'const
   ↪ ' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[62, 13]: Identifier 'i' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[66, 13]: Identifier 'index' is never reassigned; use 'const
   ↪ ' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[67, 13]: Identifier 'i' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[55, 11]: Identifier 'moduleSingle' is never reassigned; use
   ↪  'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[50, 9]: Identifier 'moduleComponent' is never reassigned;
   ↪ use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[51, 9]: Identifier 'module' is never reassigned; use 'const
   ↪ ' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[82, 11]: Identifier 'module' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[83, 11]: Identifier 'index' is never reassigned; use 'const
   ↪ ' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[84, 11]: Identifier 'm' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[80, 9]: Identifier 'saveModules' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[89, 9]: Identifier 'course' is never reassigned; use 'const
   ↪ ' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[3, 24]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[6, 29]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[31, 27]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[42, 21]: " should be '
```

```
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[56, 18]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[60, 23]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[65, 23]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[56, 15]: == should be ===
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[60, 20]: == should be ===
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[65, 20]: == should be ===
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[30, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[56, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[60, 14]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[65, 14]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[81, 8]: missing whitespace
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[18,
   ↪  21]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[6,
   ↪ 13]: The selector of the component "StatisticsComponent" should be named
   ↪  kebab-case and include dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[25, 9]: if
   ↪  statements must be braced
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[27, 9]:
   ↪ else statements must be braced
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[39, 5]: if
   ↪  statements must be braced
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[38, 17]:
   ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[41, 9]:
   ↪ Identifier 'request' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[4, 24]: "
   ↪ should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[23, 21]: "
   ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[42, 22]: "
   ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[43, 38]:
   ↪ missing whitespace
```

```
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[43, 39]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[29, 21]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[30, 21]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[30, 35]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[30, 28]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[9, 3]: comment
    ↪ must start with a space
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[35, 21]: missing
    ↪ whitespace
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[5, 24]: " should
    ↪ be '
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[28, 21]: " should
    ↪ be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[21, 78]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[22, 46]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[28, 13]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 33]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 39]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 31]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[22, 8]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[10, 50]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 12]: "
    ↪ should be '
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 29]: "
    ↪ should be '
```

```
ERROR: /angular/src/app/register/register.component.ts[29, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[30, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[31, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[32, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[33, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[34, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[35, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[42, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[33, 17]: Type string
    ↪ trivially inferred from a string literal, remove type annotation
ERROR: /angular/src/app/register/register.component.ts[34, 16]: Type string
    ↪ trivially inferred from a string literal, remove type annotation
ERROR: /angular/src/app/register/register.component.ts[22, 4]: missing
    ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[49, 18]: missing
    ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[50, 76]: missing
    ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[58, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[58, 5]: misplaced 'else'
ERROR: /angular/src/app/register/register.component.ts[59, 23]: missing
    ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[62, 11]: missing
    ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[62, 7]: misplaced 'else'
ERROR: /angular/src/app/register/register.component.ts[51, 11]: Identifier '
    ↪ data' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/register/register.component.ts[6, 29]: " should be '
ERROR: /angular/src/app/register/register.component.ts[33, 26]: " should be '
```

```
ERROR: /angular/src/app/register/register.component.ts[34, 25]: " should be '
ERROR: /angular/src/app/register/register.component.ts[43, 28]: " should be '
ERROR: /angular/src/app/register/register.component.ts[50, 35]: " should be '
ERROR: /angular/src/app/register/register.component.ts[53, 24]: " should be '
ERROR: /angular/src/app/register/register.component.ts[60, 25]: " should be '
ERROR: /angular/src/app/register/register.component.ts[63, 25]: " should be '
ERROR: /angular/src/app/register/register.component.ts[50, 7]: missing
   ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[59, 9]: missing
   ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[38, 3]: Implement
   ↪ lifecycle hook interface OnInit for method ngOnInit in class
   ↪ RegisterComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/app.module.ts[11, 1]: Exceeds maximum line length of
   ↪ 140
ERROR: /angular/src/app/app.module.ts[25, 29]: " should be '
ERROR: /angular/src/app/app.module.ts[27, 23]: " should be '
ERROR: /angular/src/app/app.module.ts[45, 35]: " should be '
ERROR: /angular/src/app/app.module.ts[49, 32]: " should be '
ERROR: /angular/src/app/app.module.ts[69, 14]: " should be '
ERROR: /angular/src/app/app.module.ts[73, 13]: " should be '
ERROR: /angular/src/app/app.module.ts[77, 14]: " should be '
ERROR: /angular/src/app/app.module.ts[83, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[95, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[99, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[100, 21]: " should be '
ERROR: /angular/src/app/app.module.ts[101, 20]: " should be '
ERROR: /angular/src/app/app.module.ts[104, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[108, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[112, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[118, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[125, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[129, 19]: " should be '
ERROR: /angular/src/app/app.module.ts[137, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[142, 17]: " should be '
ERROR: /angular/src/app/app.module.ts[73, 13]: missing whitespace

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: `ng lint`
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
```

```
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_13_42_674Z-debug.log
```

pep8 Output

```
learning_base/models.py:5:1: E302 expected 2 blank lines, found 1
learning_base/models.py:21:1: E302 expected 2 blank lines, found 1
learning_base/models.py:23:80: E501 line too long (88 > 79 characters)
learning_base/models.py:41:80: E501 line too long (106 > 79 characters)
learning_base/models.py:63:1: E302 expected 2 blank lines, found 1
learning_base/models.py:65:80: E501 line too long (85 > 79 characters)
learning_base/models.py:66:80: E501 line too long (84 > 79 characters)
learning_base/models.py:83:80: E501 line too long (104 > 79 characters)
learning_base/models.py:95:5: E265 block comment should start with '# '
learning_base/models.py:117:14: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/models.py:117:16: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/serializers.py:22:80: E501 line too long (96 > 79 characters)
learning_base/serializers.py:31:80: E501 line too long (92 > 79 characters)
learning_base/serializers.py:50:80: E501 line too long (95 > 79 characters)
learning_base/serializers.py:56:80: E501 line too long (82 > 79 characters)
learning_base/views.py:10:80: E501 line too long (80 > 79 characters)
learning_base/views.py:21:1: E302 expected 2 blank lines, found 1
learning_base/views.py:36:80: E501 line too long (100 > 79 characters)
learning_base/views.py:42:1: E302 expected 2 blank lines, found 1
learning_base/views.py:51:25: E231 missing whitespace after ','
learning_base/views.py:55:80: E501 line too long (100 > 79 characters)
learning_base/views.py:56:80: E501 line too long (103 > 79 characters)
learning_base/views.py:71:5: E265 block comment should start with '# '
learning_base/views.py:74:1: E302 expected 2 blank lines, found 1
learning_base/views.py:79:48: E703 statement ends with a semicolon
learning_base/views.py:81:15: E712 comparison to False should be 'if cond is
    ↪ False:' or 'if not cond:'
learning_base/views.py:88:1: E302 expected 2 blank lines, found 1
learning_base/views.py:93:1: E302 expected 2 blank lines, found 1
learning_base/views.py:96:13: E711 comparison to None should be 'if cond is
    ↪ None:'
learning_base/views.py:109:25: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:109:27: E251 unexpected spaces around keyword /
    ↪ parameter equals
```

```
learning_base/views.py:109:52: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:109:54: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:119:80: E501 line too long (102 > 79 characters)
learning_base/views.py:124:29: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:124:31: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:124:57: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:124:59: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:128:9: E265 block comment should start with '# '
learning_base/views.py:131:80: E501 line too long (85 > 79 characters)
learning_base/views.py:141:61: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:141:63: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:142:80: E501 line too long (81 > 79 characters)
learning_base/views.py:171:48: E703 statement ends with a semicolon
learning_base/views.py:173:15: E712 comparison to False should be 'if cond is
    ↪ False:' or 'if not cond:'
learning_base/views.py:194:80: E501 line too long (99 > 79 characters)
learning_base/migrations/0001_initial.py:21:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:22:80: E501 line too long (215 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:23:80: E501 line too long (149 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:24:80: E501 line too long (280 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:25:80: E501 line too long (105 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:31:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:32:80: E501 line too long (113 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:38:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:39:80: E501 line too long (136 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:40:80: E501 line too long (128 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:41:80: E501 line too long (219 > 79
    ↪ characters)
```

```
learning_base/migrations/0001_initial.py:50:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:51:80: E501 line too long (101 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:52:80: E501 line too long (106 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:58:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:59:80: E501 line too long (131 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:60:80: E501 line too long (134 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:61:80: E501 line too long (146 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:62:80: E501 line too long (176 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:71:80: E501 line too long (201 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:72:80: E501 line too long (93 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:82:80: E501 line too long (195 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:87:80: E501 line too long (81 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:92:80: E501 line too long (99 > 79
    ↪ characters)
learning_base/question/models.py:4:1: E302 expected 2 blank lines, found 1
learning_base/question/models.py:6:80: E501 line too long (87 > 79 characters)
learning_base/question/models.py:7:80: E501 line too long (83 > 79 characters)
learning_base/question/serializer.py:6:1: E302 expected 2 blank lines, found 1
learning_base/question/serializer.py:13:66: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:13:68: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:13:80: E501 line too long (104 > 79
    ↪ characters)
learning_base/question/serializer.py:21:1: E302 expected 2 blank lines, found 1
learning_base/question/serializer.py:28:73: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:28:75: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:28:80: E501 line too long (111 > 79
    ↪ characters)
learning_base/question/serializer.py:31:80: E501 line too long (80 > 79
    ↪ characters)
```

```
learning_base/question/multiply_choice/models.py:4:1: E302 expected 2 blank
    ↪ lines, found 1
learning_base/question/multiply_choice/models.py:41:80: E501 line too long (104
    ↪  > 79 characters)
learning_base/question/multiply_choice/models.py:56:21: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:56:23: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:57:19: E201 whitespace after
    ↪ '('
learning_base/question/multiply_choice/models.py:57:61: E202 whitespace before
    ↪ ')'
learning_base/question/multiply_choice/models.py:59:22: E225 missing whitespace
    ↪  around operator
learning_base/question/multiply_choice/models.py:67:70: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:67:72: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:67:80: E501 line too long (85
    ↪ > 79 characters)
learning_base/question/multiply_choice/serializer.py:4:1: E302 expected 2 blank
    ↪  lines, found 1
learning_base/question/multiply_choice/serializer.py:18:1: E302 expected 2
    ↪ blank lines, found 1
learning_base/question/multiply_choice/serializer.py:19:51: E251 unexpected
    ↪ spaces around keyword / parameter equals
learning_base/question/multiply_choice/serializer.py:19:53: E251 unexpected
    ↪ spaces around keyword / parameter equals
```

coverage Output

Coverage report: 74%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	11	0	0	100%
django/learning_base/drag_and_drop/__init__.py	0	0	0	100%
django/learning_base/drag_and_drop/models.py	4	0	0	100%
django/learning_base/drag_and_drop/serializer.py	4	0	0	100%
django/learning_base/models.py	106	20	0	81%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	37	11	0	70%
django/learning_base/multiple_choice/serializer.py	53	21	0	60%
django/learning_base/serializers.py	192	60	0	69%
django/learning_base/tests.py	159	0	0	100%
django/learning_base/views.py	226	99	0	56%
django/manage.py	13	6	0	54%
Total	**837**	**217**	**0**	**74%**

coverage.py v4.4.1, created at 2017-09-09 15:36

Security

bandit Output

```
Run started:2017-06-28 10:27:29.278420

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 1077
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.2 Iteration 8 - 06.07.

Abgegebene Userstories: 12

Commit Hash: cf525f64091e6c892b7e471fc8048bd8c056d69e

Checklisten

Veränderbarkeit

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
```

```
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
npm info lifecycle clonecademy@0.0.0~lint: clonecademy@0.0.0
[..................] - : info lifecycle clonecademy@0.0.0~lint: clonecademy@0.0

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/app.component.ts[10, 16]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[64, 5]: if statements must be
   ↪ braced
ERROR: /angular/src/app/login/login.component.ts[71, 1]: space indentation
   ↪ expected
ERROR: /angular/src/app/login/login.component.ts[72, 1]: space indentation
   ↪ expected
ERROR: /angular/src/app/login/login.component.ts[42, 17]: Type string trivially
   ↪  inferred from a string literal, remove type annotation
ERROR: /angular/src/app/login/login.component.ts[43, 16]: Type string trivially
   ↪  inferred from a string literal, remove type annotation
ERROR: /angular/src/app/login/login.component.ts[45, 13]: Type string trivially
   ↪  inferred from a string literal, remove type annotation
ERROR: /angular/src/app/login/login.component.ts[57, 10]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[63, 13]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[80, 41]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[66, 9]: Identifier '
   ↪ newUserInfo' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/login/login.component.ts[6, 24]: " should be '
ERROR: /angular/src/app/login/login.component.ts[17, 14]: " should be '
ERROR: /angular/src/app/login/login.component.ts[18, 18]: " should be '
ERROR: /angular/src/app/login/login.component.ts[20, 14]: " should be '
ERROR: /angular/src/app/login/login.component.ts[21, 18]: " should be '
ERROR: /angular/src/app/login/login.component.ts[42, 26]: " should be '
ERROR: /angular/src/app/login/login.component.ts[43, 25]: " should be '
ERROR: /angular/src/app/login/login.component.ts[45, 22]: " should be '
ERROR: /angular/src/app/login/login.component.ts[73, 22]: " should be '
ERROR: /angular/src/app/login/login.component.ts[60, 47]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[72, 25]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[80, 7]: missing whitespace
ERROR: /angular/src/app/course/course.component.ts[17, 14]: Type boolean
   ↪ trivially inferred from a boolean literal, remove type annotation
ERROR: /angular/src/app/course/course.component.ts[32, 57]: missing whitespace
ERROR: /angular/src/app/course/course.component.ts[30, 13]: Identifier '
   ↪ lastModule' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/course/course.component.ts[31, 13]: Identifier '
   ↪ lastQuestion' is never reassigned; use 'const' instead of 'let'.
```

```
ERROR: /angular/src/app/course/course.component.ts[24, 42]: " should be '
ERROR: /angular/src/app/course/course.component.ts[24, 31]: missing whitespace
ERROR: /angular/src/app/course/course.component.ts[24, 32]: missing whitespace
ERROR: /angular/src/app/course/course.component.ts[32, 11]: missing whitespace
ERROR: /angular/src/app/add-question/add-question.module.ts[9, 21]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-question/add-question.module.ts[10, 12]: " should
    ↪ be '
ERROR: /angular/src/app/add-question/add-question.module.ts[8, 14]: The name of
    ↪  the class AddQuestionModule should end with the suffix Component (https
    ↪ ://goo.gl/5X1TE7)
ERROR: /angular/src/app/add-question/add-question.component.ts[1, 1]: Exceeds
    ↪ maximum line length of 140
ERROR: /angular/src/app/add-question/add-question.component.ts[30, 16]: missing
    ↪  whitespace
ERROR: /angular/src/app/add-question/add-question.component.ts[37, 11]: missing
    ↪  whitespace
ERROR: /angular/src/app/add-question/add-question.component.ts[41, 14]: missing
    ↪  whitespace
ERROR: /angular/src/app/add-question/add-question.component.ts[44, 26]: missing
    ↪  whitespace
ERROR: /angular/src/app/add-question/add-question.component.ts[32, 9]:
    ↪ Identifier 'question' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/add-question/add-question.component.ts[42, 9]:
    ↪ Identifier 'response' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/add-question/add-question.component.ts[38, 23]: "
    ↪ should be '
ERROR: /angular/src/app/add-question/add-question.component.ts[44, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-multiply-choice/add-multiply-choice.component.ts
    ↪ [20, 14]: missing whitespace
ERROR: /angular/src/app/add-multiply-choice/add-multiply-choice.component.ts
    ↪ [28, 30]: missing whitespace
ERROR: /angular/src/app/add-multiply-choice/add-multiply-choice.component.ts
    ↪ [32, 14]: missing whitespace
ERROR: /angular/src/app/add-multiply-choice/add-multiply-choice.component.ts[3,
    ↪  35]: " should be '
ERROR: /angular/src/app/add-multiply-choice/add-multiply-choice.component.ts
    ↪ [17, 28]: " should be '
ERROR: /angular/src/app/add-multiply-choice/add-multiply-choice.component.ts
    ↪ [22, 13]: " should be '
ERROR: /angular/src/app/add-multiply-choice/add-multiply-choice.component.ts
    ↪ [33, 30]: " should be '
ERROR: /angular/src/app/add-multiply-choice/add-multiply-choice.component.ts
    ↪ [33, 42]: missing whitespace
```

```
ERROR: /angular/src/app/add-multiply-choice/add-multiply-choice.component.ts
    ↪ [16, 3]: Implement lifecycle hook interface OnInit for method ngOnInit
    ↪ in class AddMultiplyChoiceComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/add-module/add-module.component.ts[1, 1]: Exceeds
    ↪ maximum line length of 140
ERROR: /angular/src/app/add-module/add-module.component.ts[35, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[36, 40]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[51, 10]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[55, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[59, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[63, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[65, 55]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[68, 21]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[39, 11]: Identifier
    ↪ 'question' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/add-module/add-module.component.ts[40, 11]: Identifier
    ↪ 'q' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/add-module/add-module.component.ts[69, 13]: Identifier
    ↪ 'save' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/add-module/add-module.component.ts[66, 11]: Identifier
    ↪ 'tmp' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/add-module/add-module.component.ts[67, 11]: Identifier
    ↪ 'index' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/add-module/add-module.component.ts[64, 9]: Identifier '
    ↪ values' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/add-module/add-module.component.ts[3, 38]: " should be
    ↪ '
ERROR: /angular/src/app/add-module/add-module.component.ts[7, 44]: " should be
    ↪ '
ERROR: /angular/src/app/add-module/add-module.component.ts[17, 12]: " should be
    ↪  '
ERROR: /angular/src/app/add-module/add-module.component.ts[52, 21]: " should be
    ↪  '
ERROR: /angular/src/app/add-module/add-module.component.ts[56, 21]: " should be
    ↪  '
ERROR: /angular/src/app/add-module/add-module.component.ts[60, 21]: " should be
    ↪  '
```

```
ERROR: /angular/src/app/add-module/add-module.component.ts[36, 27]: != should
    ↪ be !==
ERROR: /angular/src/app/add-module/add-module.component.ts[36, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[65, 8]: missing
    ↪ whitespace
ERROR: /angular/src/app/add-module/add-module.component.ts[68, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[1, 1]: Exceeds
    ↪  maximum line length of 140
ERROR: /angular/src/app/create-course/create-course.component.ts[31, 14]:
    ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[38, 27]:
    ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[42, 7]:
    ↪ misplaced 'else'
ERROR: /angular/src/app/create-course/create-course.component.ts[42, 28]:
    ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[47, 7]:
    ↪ misplaced 'else'
ERROR: /angular/src/app/create-course/create-course.component.ts[47, 30]:
    ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[56, 17]:
    ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[61, 9]:
    ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[64, 53]:
    ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[43, 13]:
    ↪ Identifier 'index' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[44, 13]:
    ↪ Identifier 'i' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[48, 13]:
    ↪ Identifier 'index' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[49, 13]:
    ↪ Identifier 'i' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[37, 11]:
    ↪ Identifier 'moduleSingle' is never reassigned; use 'const' instead of '
    ↪ let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[32, 9]:
    ↪ Identifier 'moduleComponent' is never reassigned; use 'const' instead of
    ↪  'let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[33, 9]:
    ↪ Identifier 'module' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[65, 11]:
    ↪ Identifier 'module' is never reassigned; use 'const' instead of 'let'.
```

```
ERROR: /angular/src/app/create-course/create-course.component.ts[66, 11]:
   ↪ Identifier 'index' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[67, 11]:
   ↪ Identifier 'm' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[63, 9]:
   ↪ Identifier 'saveModules' is never reassigned; use 'const' instead of '
   ↪ let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[72, 9]:
   ↪ Identifier 'course' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/create-course/create-course.component.ts[3, 24]: "
   ↪ should be '
ERROR: /angular/src/app/create-course/create-course.component.ts[28, 21]: "
   ↪ should be '
ERROR: /angular/src/app/create-course/create-course.component.ts[38, 18]: "
   ↪ should be '
ERROR: /angular/src/app/create-course/create-course.component.ts[42, 23]: "
   ↪ should be '
ERROR: /angular/src/app/create-course/create-course.component.ts[47, 23]: "
   ↪ should be '
ERROR: /angular/src/app/create-course/create-course.component.ts[38, 15]: ==
   ↪ should be ===
ERROR: /angular/src/app/create-course/create-course.component.ts[42, 20]: ==
   ↪ should be ===
ERROR: /angular/src/app/create-course/create-course.component.ts[47, 20]: ==
   ↪ should be ===
ERROR: /angular/src/app/create-course/create-course.component.ts[38, 9]:
   ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[42, 14]:
   ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[47, 14]:
   ↪ missing whitespace
ERROR: /angular/src/app/create-course/create-course.component.ts[64, 8]:
   ↪ missing whitespace
ERROR: /angular/src/app/dashboard/dashboard.component.ts[1, 1]: Exceeds maximum
   ↪  line length of 140
ERROR: /angular/src/app/dashboard/dashboard.component.ts[56, 18]: missing
   ↪ whitespace
ERROR: /angular/src/app/dashboard/dashboard.component.ts[61, 28]: missing
   ↪ whitespace
ERROR: /angular/src/app/dashboard/dashboard.component.ts[63, 9]: Identifier '
   ↪ courseView' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/dashboard/dashboard.component.ts[64, 9]: Identifier '
   ↪ course' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/dashboard/dashboard.component.ts[17, 14]: " should be '
ERROR: /angular/src/app/dashboard/dashboard.component.ts[18, 18]: " should be '
ERROR: /angular/src/app/dashboard/dashboard.component.ts[20, 14]: " should be '
```

```
ERROR: /angular/src/app/dashboard/dashboard.component.ts[21, 18]: " should be '
ERROR: /angular/src/app/dashboard/dashboard.component.ts[48, 21]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 41]: missing
   ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[2, 26]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[3, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[15, 34]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[16, 29]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 7]: missing
   ↪ whitespace
ERROR: /angular/src/app/directive/module.directive.ts[10, 29]: missing
   ↪ whitespace
ERROR: /angular/src/app/question/question.module.ts[10, 46]: missing whitespace
ERROR: /angular/src/app/question/question.module.ts[20, 16]: missing whitespace
ERROR: /angular/src/app/question/question.module.ts[23, 13]: missing whitespace
ERROR: /angular/src/app/question/question.module.ts[2, 31]: " should be '
ERROR: /angular/src/app/question/question.module.ts[21, 12]: " should be '
ERROR: /angular/src/app/question/question.module.ts[7, 13]: The selector of the
   ↪  component "QuestionModule" should be named kebab-case and include dash
   ↪ (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/question/question.module.ts[10, 14]: The name of the
   ↪ class QuestionModule should end with the suffix Component (https://goo.
   ↪ gl/5X1TE7)
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
   ↪ component.ts[36, 7]: comment must start with a space
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
   ↪ component.ts[29, 16]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
   ↪ component.ts[32, 40]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
   ↪ component.ts[23, 15]: Identifier 'ans' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
   ↪ component.ts[31, 14]: Identifier 'ans' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
   ↪ component.ts[30, 9]: Identifier 'sendAnswer' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
   ↪ component.ts[3, 32]: " should be '
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
   ↪ component.ts[4, 31]: " should be '
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
   ↪ component.ts[20, 21]: " should be '
```

```
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[20, 46]: " should be '
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[20, 69]: " should be '
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[20, 49]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[20, 50]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[20, 45]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[20, 46]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[20, 31]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[20, 32]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[23, 10]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[32, 9]: missing whitespace
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[7, 13]: The selector of the component "
    ↪ MultipleChoiceQuestionComponent" should be named kebab-case and include
    ↪ dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/multiple-choice-question/multiple-choice-question.
    ↪ component.ts[16, 3]: Implement lifecycle hook interface OnInit for
    ↪ method ngOnInit in class MultipleChoiceQuestionComponent (https://goo.gl
    ↪ /w1Nwk3)
ERROR: /angular/src/app/question/question.component.ts[1, 1]: Exceeds maximum
    ↪ line length of 140
ERROR: /angular/src/app/question/question.component.ts[34, 1]: Exceeds maximum
    ↪ line length of 140
ERROR: /angular/src/app/question/question.component.ts[71, 13]: Shadowed
    ↪ variable: 'data'
ERROR: /angular/src/app/question/question.component.ts[38, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[48, 17]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[67, 11]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[75, 22]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[79, 25]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[82, 9]: missing
    ↪ whitespace
```

```
ERROR: /angular/src/app/question/question.component.ts[82, 5]: misplaced 'else'
ERROR: /angular/src/app/question/question.component.ts[87, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[90, 27]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[93, 5]: misplaced 'else'
ERROR: /angular/src/app/question/question.component.ts[93, 30]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[97, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[97, 5]: misplaced 'else'
ERROR: /angular/src/app/question/question.component.ts[60, 11]: Identifier '
    ↪ question' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/question/question.component.ts[69, 9]: Identifier 'data
    ↪ ' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/question/question.component.ts[3, 31]: " should be '
ERROR: /angular/src/app/question/question.component.ts[5, 49]: " should be '
ERROR: /angular/src/app/question/question.component.ts[7, 32]: " should be '
ERROR: /angular/src/app/question/question.component.ts[49, 21]: " should be '
ERROR: /angular/src/app/question/question.component.ts[49, 46]: " should be '
ERROR: /angular/src/app/question/question.component.ts[49, 69]: " should be '
ERROR: /angular/src/app/question/question.component.ts[70, 22]: " should be '
ERROR: /angular/src/app/question/question.component.ts[70, 47]: " should be '
ERROR: /angular/src/app/question/question.component.ts[70, 70]: " should be '
ERROR: /angular/src/app/question/question.component.ts[83, 23]: " should be '
ERROR: /angular/src/app/question/question.component.ts[99, 33]: " should be '
ERROR: /angular/src/app/question/question.component.ts[102, 31]: " should be '
ERROR: /angular/src/app/question/question.component.ts[102, 56]: " should be '
ERROR: /angular/src/app/question/question.component.ts[102, 70]: " should be '
ERROR: /angular/src/app/question/question.component.ts[49, 49]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[49, 50]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[49, 45]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[49, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[49, 31]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[49, 32]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[70, 50]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[70, 51]: missing
    ↪ whitespace
```

```
ERROR: /angular/src/app/question/question.component.ts[70, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[70, 47]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[70, 32]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[70, 33]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[102, 59]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[102, 55]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[102, 56]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[102, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[102, 42]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[79, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[90, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/question/question.component.ts[93, 12]: missing
    ↪ whitespace
ERROR: /angular/src/app/module/module.component.ts[34, 13]: missing whitespace
ERROR: /angular/src/app/module/module.component.ts[9, 35]: " should be '
ERROR: /angular/src/app/module/module.component.ts[10, 49]: " should be '
ERROR: /angular/src/app/module/module.component.ts[20, 5]: " should be '
ERROR: /angular/src/app/module/module.component.ts[36, 21]: " should be '
ERROR: /angular/src/app/module/module.component.ts[36, 46]: " should be '
ERROR: /angular/src/app/module/module.component.ts[36, 49]: missing whitespace
ERROR: /angular/src/app/module/module.component.ts[36, 50]: missing whitespace
ERROR: /angular/src/app/module/module.component.ts[36, 45]: missing whitespace
ERROR: /angular/src/app/module/module.component.ts[36, 46]: missing whitespace
ERROR: /angular/src/app/module/module.component.ts[36, 31]: missing whitespace
ERROR: /angular/src/app/module/module.component.ts[36, 32]: missing whitespace
ERROR: /angular/src/app/module/module.component.ts[28, 14]: missing whitespace
ERROR: /angular/src/app/personal_statistics/statistics.component.ts[17, 21]: "
    ↪ should be '
ERROR: /angular/src/app/personal_statistics/statistics.component.ts[6, 13]: The
    ↪  selector of the component "StatisticsComponent" should be named kebab-
    ↪ case and include dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/request-mod/request-mod.component.ts[24, 9]: if
    ↪ statements must be braced
ERROR: /angular/src/app/request-mod/request-mod.component.ts[26, 9]: else
    ↪ statements must be braced
```

```
ERROR: /angular/src/app/request-mod/request-mod.component.ts[34, 5]: if
    ↪ statements must be braced
ERROR: /angular/src/app/request-mod/request-mod.component.ts[33, 17]: missing
    ↪ whitespace
ERROR: /angular/src/app/request-mod/request-mod.component.ts[36, 9]: Identifier
    ↪  'request' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/request-mod/request-mod.component.ts[4, 24]: " should
    ↪ be '
ERROR: /angular/src/app/request-mod/request-mod.component.ts[22, 21]: " should
    ↪ be '
ERROR: /angular/src/app/request-mod/request-mod.component.ts[37, 22]: " should
    ↪ be '
ERROR: /angular/src/app/request-mod/request-mod.component.ts[39, 39]: missing
    ↪ whitespace
ERROR: /angular/src/app/request-mod/request-mod.component.ts[39, 40]: missing
    ↪ whitespace
ERROR: /angular/src/app/user-detail/user-detail.component.ts[19, 21]: missing
    ↪ whitespace
ERROR: /angular/src/app/user-detail/user-detail.component.ts[27, 19]: missing
    ↪ whitespace
ERROR: /angular/src/app/user-detail/user-detail.component.ts[28, 9]: Identifier
    ↪  'usernameJSON' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/user-detail/user-detail.component.ts[20, 21]: " should
    ↪ be '
ERROR: /angular/src/app/user-detail/user-detail.component.ts[20, 35]: " should
    ↪ be '
ERROR: /angular/src/app/user-detail/user-detail.component.ts[29, 22]: " should
    ↪ be '
ERROR: /angular/src/app/user-detail/user-detail.component.ts[20, 28]: missing
    ↪ whitespace
ERROR: /angular/src/app/profiles/profiles.component.ts[9, 3]: comment must
    ↪ start with a space
ERROR: /angular/src/app/profiles/profiles.component.ts[5, 24]: " should be '
ERROR: /angular/src/app/profiles/profiles.component.ts[30, 21]: " should be '
ERROR: /angular/src/app/profile-page/profile-page.component.ts[21, 5]: unused
    ↪ expression, expected an assignment or function call
ERROR: /angular/src/app/profile-page/profile-page.component.ts[29, 17]: missing
    ↪  whitespace
ERROR: /angular/src/app/profile-page/profile-page.component.ts[22, 21]: "
    ↪ should be '
ERROR: /angular/src/app/app.module.ts[33, 35]: " should be '
ERROR: /angular/src/app/app.module.ts[34, 32]: " should be '
ERROR: /angular/src/app/app.module.ts[59, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[63, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[67, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[71, 11]: " should be '
```

```
ERROR: /angular/src/app/app.module.ts[75, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[79, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[83, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[87, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[91, 11]: " should be '

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: `ng lint`
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_15_17_024Z-debug.log
```

pep8 Output

```
learning_base/models.py:5:1: E302 expected 2 blank lines, found 1
learning_base/models.py:21:1: E302 expected 2 blank lines, found 1
learning_base/models.py:23:80: E501 line too long (88 > 79 characters)
learning_base/models.py:41:80: E501 line too long (106 > 79 characters)
learning_base/models.py:63:1: E302 expected 2 blank lines, found 1
learning_base/models.py:65:80: E501 line too long (85 > 79 characters)
learning_base/models.py:66:80: E501 line too long (84 > 79 characters)
learning_base/models.py:83:80: E501 line too long (104 > 79 characters)
learning_base/models.py:95:5: E265 block comment should start with '# '
learning_base/models.py:117:14: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/models.py:117:16: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/serializers.py:22:80: E501 line too long (96 > 79 characters)
learning_base/serializers.py:31:80: E501 line too long (92 > 79 characters)
learning_base/serializers.py:50:80: E501 line too long (95 > 79 characters)
learning_base/serializers.py:56:80: E501 line too long (82 > 79 characters)
learning_base/views.py:10:80: E501 line too long (80 > 79 characters)
learning_base/views.py:21:1: E302 expected 2 blank lines, found 1
learning_base/views.py:36:80: E501 line too long (100 > 79 characters)
learning_base/views.py:42:1: E302 expected 2 blank lines, found 1
learning_base/views.py:51:25: E231 missing whitespace after ','
learning_base/views.py:55:80: E501 line too long (100 > 79 characters)
```

```
learning_base/views.py:56:80: E501 line too long (103 > 79 characters)
learning_base/views.py:71:5: E265 block comment should start with '# '
learning_base/views.py:74:1: E302 expected 2 blank lines, found 1
learning_base/views.py:79:48: E703 statement ends with a semicolon
learning_base/views.py:81:15: E712 comparison to False should be 'if cond is
    ↪ False:' or 'if not cond:'
learning_base/views.py:88:1: E302 expected 2 blank lines, found 1
learning_base/views.py:93:1: E302 expected 2 blank lines, found 1
learning_base/views.py:96:13: E711 comparison to None should be 'if cond is
    ↪ None:'
learning_base/views.py:109:25: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:109:27: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:109:52: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:109:54: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:119:80: E501 line too long (102 > 79 characters)
learning_base/views.py:124:29: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:124:31: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:124:57: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:124:59: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:128:9: E265 block comment should start with '# '
learning_base/views.py:131:80: E501 line too long (85 > 79 characters)
learning_base/views.py:141:61: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:141:63: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:142:80: E501 line too long (81 > 79 characters)
learning_base/views.py:171:48: E703 statement ends with a semicolon
learning_base/views.py:173:15: E712 comparison to False should be 'if cond is
    ↪ False:' or 'if not cond:'
learning_base/views.py:194:80: E501 line too long (99 > 79 characters)
learning_base/migrations/0001_initial.py:21:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:22:80: E501 line too long (215 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:23:80: E501 line too long (149 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:24:80: E501 line too long (280 > 79
    ↪ characters)
```

```
learning_base/migrations/0001_initial.py:25:80: E501 line too long (105 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:31:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:32:80: E501 line too long (113 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:38:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:39:80: E501 line too long (136 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:40:80: E501 line too long (128 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:41:80: E501 line too long (219 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:50:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:51:80: E501 line too long (101 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:52:80: E501 line too long (106 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:58:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:59:80: E501 line too long (131 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:60:80: E501 line too long (134 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:61:80: E501 line too long (146 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:62:80: E501 line too long (176 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:71:80: E501 line too long (201 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:72:80: E501 line too long (93 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:82:80: E501 line too long (195 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:87:80: E501 line too long (81 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:92:80: E501 line too long (99 > 79
    ↪ characters)
learning_base/question/models.py:4:1: E302 expected 2 blank lines, found 1
learning_base/question/models.py:6:80: E501 line too long (87 > 79 characters)
learning_base/question/models.py:7:80: E501 line too long (83 > 79 characters)
learning_base/question/serializer.py:6:1: E302 expected 2 blank lines, found 1
learning_base/question/serializer.py:13:66: E251 unexpected spaces around
    ↪ keyword / parameter equals
```

```
learning_base/question/serializer.py:13:68: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:13:80: E501 line too long (104 > 79
    ↪ characters)
learning_base/question/serializer.py:21:1: E302 expected 2 blank lines, found 1
learning_base/question/serializer.py:28:73: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:28:75: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:28:80: E501 line too long (111 > 79
    ↪ characters)
learning_base/question/serializer.py:31:80: E501 line too long (80 > 79
    ↪ characters)
learning_base/question/multiply_choice/models.py:4:1: E302 expected 2 blank
    ↪ lines, found 1
learning_base/question/multiply_choice/models.py:41:80: E501 line too long (104
    ↪  > 79 characters)
learning_base/question/multiply_choice/models.py:56:21: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:56:23: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:57:19: E201 whitespace after
    ↪ '('
learning_base/question/multiply_choice/models.py:57:61: E202 whitespace before
    ↪ ')'
learning_base/question/multiply_choice/models.py:59:22: E225 missing whitespace
    ↪  around operator
learning_base/question/multiply_choice/models.py:67:70: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:67:72: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:67:80: E501 line too long (85
    ↪ > 79 characters)
learning_base/question/multiply_choice/serializer.py:4:1: E302 expected 2 blank
    ↪  lines, found 1
learning_base/question/multiply_choice/serializer.py:18:1: E302 expected 2
    ↪ blank lines, found 1
learning_base/question/multiply_choice/serializer.py:19:51: E251 unexpected
    ↪ spaces around keyword / parameter equals
learning_base/question/multiply_choice/serializer.py:19:53: E251 unexpected
    ↪ spaces around keyword / parameter equals
```

Datensicherheit

bandit Output

```
Run started:2017-09-12 10:27:39.861074

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 1171
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.3 Iteration 9 - 13.07.

Abgegebene Userstories: keine

Commit Hash: e2b7fed9dbc0e929bb28e7ccf30cd68aa0d73c19

Checklisten

Veränderbarkeit

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
npm info lifecycle clonecademy@0.0.0~lint: clonecademy@0.0.0
```

```
> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/error-message/error-message.component.ts[20, 3]:
    ↪ Implement lifecycle hook interface OnInit for method ngOnInit in class
    ↪ ErrorMessageComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/service/error.service.ts[12, 21]: missing whitespace
ERROR: /angular/src/app/service/error.service.ts[13, 11]: Identifier 'dialogRef
    ↪ ' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/loader/loader.component.ts[10, 42]: expected nospace
    ↪ before colon in parameter
ERROR: /angular/src/app/loader/loader.component.ts[12, 3]: Implement lifecycle
    ↪ hook interface AfterViewInit for method ngAfterViewInit in class
    ↪ LoaderComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/service/sass-helper/sass-helper.ts[20, 13]: Identifier
    ↪ 'bodyStyles' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/sass-helper/sass-helper.ts[12, 37]: missing
    ↪ whitespace
ERROR: /angular/src/app/service/sass-helper/sass-helper.ts[6, 15]: The selector
    ↪  of the component "SassHelperComponent" should have prefix "app" (https
    ↪ ://goo.gl/cix8BY)
ERROR: /angular/src/app/app.component.ts[14, 78]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[38, 1]: Exceeds maximum line
    ↪ length of 140
ERROR: /angular/src/app/login/login.component.ts[42, 14]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[43, 19]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[50, 9]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[50, 5]: misplaced 'else'
ERROR: /angular/src/app/login/login.component.ts[56, 41]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[47, 13]: Identifier 'dialogRef
    ↪ ' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/login/login.component.ts[5, 29]: " should be '
ERROR: /angular/src/app/login/login.component.ts[8, 24]: " should be '
ERROR: /angular/src/app/login/login.component.ts[21, 14]: " should be '
ERROR: /angular/src/app/login/login.component.ts[22, 18]: " should be '
ERROR: /angular/src/app/login/login.component.ts[24, 14]: " should be '
ERROR: /angular/src/app/login/login.component.ts[25, 18]: " should be '
ERROR: /angular/src/app/login/login.component.ts[51, 29]: " should be '
ERROR: /angular/src/app/login/login.component.ts[38, 79]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[43, 7]: missing whitespace
ERROR: /angular/src/app/login/login.component.ts[56, 7]: missing whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[45, 9]: comment
    ↪ must start with a space
```

```
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[28, 32]: missing
    ↪  whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[42, 40]: missing
    ↪  whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[44, 20]: missing
    ↪  whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[43, 9]:
    ↪ Identifier 'percent' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[15, 30]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[15, 40]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[16, 30]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[16, 40]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[48, 12]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[44, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[2, 26]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[3, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[15, 34]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[16, 29]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/menu/menu.component.ts[16, 72]: missing whitespace
ERROR: /angular/src/app/menu/menu.component.ts[17, 34]: " should be '
ERROR: /angular/src/app/learn/course/course.component.ts[25, 5]: comment must
    ↪ start with a space
ERROR: /angular/src/app/learn/course/course.component.ts[19, 14]: Type boolean
    ↪ trivially inferred from a boolean literal, remove type annotation
ERROR: /angular/src/app/learn/course/course.component.ts[28, 23]: Type string
    ↪ trivially inferred from a string literal, remove type annotation
ERROR: /angular/src/app/learn/course/course.component.ts[45, 14]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[62, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[88, 36]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[95, 9]: misplaced '
    ↪ else'
```

```
ERROR: /angular/src/app/learn/course/course.component.ts[91, 42]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[98, 28]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[99, 55]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[100, 70]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[101, 54]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[90, 15]: Identifier '
    ↪ lastQuestion' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course/course.component.ts[87, 13]: Identifier '
    ↪ lastModule' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course/course.component.ts[57, 22]: " should be '
ERROR: /angular/src/app/learn/course/course.component.ts[77, 19]: " should be '
ERROR: /angular/src/app/learn/course/course.component.ts[79, 44]: " should be '
ERROR: /angular/src/app/learn/course/course.component.ts[111, 29]: " should be
    ↪ '
ERROR: /angular/src/app/learn/course/course.component.ts[88, 23]: != should be
    ↪ !==
ERROR: /angular/src/app/learn/course/course.component.ts[91, 34]: == should be
    ↪ ===
ERROR: /angular/src/app/learn/course/course.component.ts[79, 33]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[79, 34]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[83, 65]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[83, 66]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[102, 37]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[102, 38]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[102, 42]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[102, 43]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[26, 25]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[27, 23]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[28, 23]: missing
    ↪ whitespace
```

```
ERROR: /angular/src/app/learn/course/course.component.ts[29, 24]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[32, 23]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[32, 28]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[88, 11]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[91, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[98, 11]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[99, 14]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[100, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course/course.component.ts[101, 17]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/module.directive.ts[10, 29]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[10, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[24, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[28, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[31, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[2, 31]: " should be '
ERROR: /angular/src/app/learn/question/question.module.ts[25, 12]: " should be
    ↪ '
ERROR: /angular/src/app/learn/question/question.module.ts[7, 13]: The selector
    ↪ of the component "QuestionModule" should be named kebab-case and include
    ↪  dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/question/question.module.ts[10, 14]: The name of
    ↪ the class QuestionModule should end with the suffix Component (https://
    ↪ goo.gl/5X1TE7)
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[29, 7]: comment must start with a space
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[22, 16]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[25, 20]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[24, 14]: Identifier 'ans' is never reassigned; use 'const'
    ↪  instead of 'let'.
```

```
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[23, 9]: Identifier 'sendAnswer' is never reassigned; use '
    ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[3, 32]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[4, 31]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[25, 9]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[7, 13]: The selector of the component "
    ↪ MultipleChoiceQuestionComponent" should be named kebab-case and include
    ↪ dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[16, 3]: Implement lifecycle hook interface OnInit for
    ↪ method ngOnInit in class MultipleChoiceQuestionComponent (https://goo.gl
    ↪ /w1Nwk3)
ERROR: /angular/src/app/learn/question/question.component.ts[1, 1]: Exceeds
    ↪ maximum line length of 140
ERROR: /angular/src/app/learn/question/question.component.ts[89, 13]: Shadowed
    ↪ variable: 'data'
ERROR: /angular/src/app/learn/question/question.component.ts[106, 58]: Shadowed
    ↪  variable: 'data'
ERROR: /angular/src/app/learn/question/question.component.ts[112, 79]: Shadowed
    ↪  variable: 'data'
ERROR: /angular/src/app/learn/question/question.component.ts[59, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[63, 17]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[85, 11]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[93, 22]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[98, 27]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[109, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[109, 5]: misplaced
    ↪  'else'
ERROR: /angular/src/app/learn/question/question.component.ts[102, 33]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[105, 11]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[105, 7]: misplaced
    ↪  'else'
```

```
ERROR: /angular/src/app/learn/question/question.component.ts[118, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[119, 27]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[122, 5]: misplaced
    ↪  'else'
ERROR: /angular/src/app/learn/question/question.component.ts[122, 30]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[126, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[126, 5]: misplaced
    ↪  'else'
ERROR: /angular/src/app/learn/question/question.component.ts[76, 11]:
    ↪ Identifier 'question' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/question/question.component.ts[87, 9]: Identifier
    ↪  'data' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/question/question.component.ts[3, 31]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[7, 49]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[9, 32]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[64, 21]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[64, 46]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[64, 83]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[88, 22]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[88, 47]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[88, 85]: " should
    ↪ be '
ERROR: /angular/src/app/learn/question/question.component.ts[102, 30]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[106, 28]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[112, 26]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[128, 33]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[131, 31]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[131, 56]: " should
    ↪  be '
```

```
ERROR: /angular/src/app/learn/question/question.component.ts[131, 80]: " should
    ↪  be '
ERROR: /angular/src/app/learn/question/question.component.ts[102, 27]: !=
    ↪ should be !==
ERROR: /angular/src/app/learn/question/question.component.ts[64, 49]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 45]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 31]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 32]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[64, 78]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[88, 50]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[88, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[88, 47]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[88, 32]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[88, 33]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[88, 79]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[88, 120]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[131, 59]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[131, 55]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[131, 56]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[131, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[131, 42]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[98, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[102, 9]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.component.ts[119, 7]: missing
    ↪ whitespace
```

```
ERROR: /angular/src/app/learn/question/question.component.ts[122, 12]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[40, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[9, 35]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[10, 49]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[20, 5]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 21]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 46]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 49]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 50]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 45]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 31]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 32]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[28, 14]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.module.ts
    ↪ [14, 25]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.module.ts
    ↪ [18, 29]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.module.ts
    ↪ [19, 24]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.module.ts
    ↪ [15, 12]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.module.ts
    ↪ [19, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.module.ts
    ↪ [8, 14]: The name of the class AddQuestionModule should end with the
    ↪ suffix Component (https://goo.gl/5X1TE7)
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[41, 7]: comment must start with a space
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[1, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[28, 13]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[33, 13]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[40, 72]: missing whitespace
```

```
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[54, 12]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[55, 38]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[61, 15]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[67, 42]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[46, 9]: Identifier 'question' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[62, 9]: Identifier 'response' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[4, 25]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[20, 18]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[55, 34]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[56, 25]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[66, 25]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[68, 30]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[55, 31]: == should be ===
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[67, 29]: == should be ===
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[55, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
    ↪ .ts[67, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[21, 18]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[24, 44]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[33, 37]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[34, 76]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[39, 33]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[43, 14]: missing whitespace
```

```
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[47, 25]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[48, 49]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[49, 37]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[23, 9]: Identifier 'answers' is never reassigned;
    ↪ use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[3, 35]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[5, 25]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[16, 83]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[28, 13]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[44, 30]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[34, 67]: == should be ===
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[24, 8]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[34, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[44, 45]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[48, 8]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
    ↪ choice.component.ts[49, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [75, 7]: comment must start with a space
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [1, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [22, 10]: Type string trivially inferred from a string literal, remove
    ↪ type annotation
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [23, 17]: Type string trivially inferred from a string literal, remove
    ↪ type annotation
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [41, 82]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [53, 83]: missing whitespace
```

```
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [55, 52]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [56, 42]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [67, 20]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [73, 10]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [78, 16]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [79, 54]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [85, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [90, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [97, 13]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [100, 55]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [103, 21]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [43, 11]: Identifier 'question' is never reassigned; use 'const' instead
    ↪  of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [44, 11]: Identifier 'q' is never reassigned; use 'const' instead of '
    ↪ let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [104, 13]: Identifier 'save' is never reassigned; use 'const' instead of
    ↪  'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [101, 11]: Identifier 'tmp' is never reassigned; use 'const' instead of
    ↪ 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [102, 11]: Identifier 'index' is never reassigned; use 'const' instead
    ↪ of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [99, 9]: Identifier 'values' is never reassigned; use 'const' instead of
    ↪  'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [3, 38]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [5, 25]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [8, 44]: " should be '
```

```
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [19, 12]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [22, 19]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [23, 26]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [79, 25]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [80, 23]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [86, 21]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [91, 21]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [79, 22]: == should be ===
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [79, 45]: == should be ===
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [55, 8]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [56, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [79, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [100, 8]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [103, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.ts
    ↪ [67, 3]: Implement lifecycle hook interface AfterViewInit for method
    ↪ ngAfterViewInit in class AddModuleComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[55, 9]: comment must start with a space
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[123, 9]: comment must start with a space
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[1, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[31, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[122, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
    ↪ component.ts[38, 3]: Declaration of instance field not allowed after
    ↪ declaration of instance method. Instead, this should come at the
    ↪ beginning of the class/interface.
```

```
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[44, 3]: Declaration of instance field not allowed after
   ↪ declaration of instance method. Instead, this should come at the
   ↪ beginning of the class/interface.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[46, 3]: Declaration of instance field not allowed after
   ↪ declaration of instance method. Instead, this should come at the
   ↪ beginning of the class/interface.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[51, 3]: Declaration of instance field not allowed after
   ↪ declaration of instance method. Instead, this should come at the
   ↪ beginning of the class/interface.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[21, 11]: Type number trivially inferred from a number
   ↪ literal, remove type annotation
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[34, 26]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[40, 26]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[53, 20]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[54, 33]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[77, 80]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[80, 22]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[83, 34]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[86, 26]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[87, 48]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[96, 27]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[101, 7]: misplaced 'else'
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[101, 28]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[106, 7]: misplaced 'else'
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[106, 30]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[115, 17]: missing whitespace
```

```
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[119, 10]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[121, 16]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[128, 20]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[130, 53]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[140, 22]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[88, 13]: Identifier 'question' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[102, 13]: Identifier 'index' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[103, 13]: Identifier 'i' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[107, 13]: Identifier 'index' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[108, 13]: Identifier 'i' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[95, 11]: Identifier 'moduleSingle' is never reassigned; use
   ↪  'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[78, 9]: Identifier 'moduleComponent' is never reassigned;
   ↪ use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[79, 9]: Identifier 'module' is never reassigned; use 'const
   ↪ ' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[122, 11]: Identifier 'course' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[120, 9]: Identifier 'saveModules' is never reassigned; use
   ↪ 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[131, 11]: Identifier 'module' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[132, 11]: Identifier 'index' is never reassigned; use '
   ↪ const' instead of 'let'.
```

```
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[133, 11]: Identifier 'm' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[129, 9]: Identifier 'saveModules' is never reassigned; use
   ↪ 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[3, 40]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[8, 29]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[26, 56]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[26, 68]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[26, 85]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[26, 97]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[31, 73]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[31, 99]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[31, 129]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[31, 160]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[47, 28]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[70, 21]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[96, 18]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[101, 23]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[106, 23]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[122, 98]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[96, 15]: == should be ===
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[101, 20]: == should be ===
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[106, 20]: == should be ===
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[54, 7]: missing whitespace
```

```
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[80, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[83, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[86, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[87, 10]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[96, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[101, 14]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[106, 14]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[121, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[130, 8]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.ts[53, 3]: Implement lifecycle hook interface AfterViewInit
   ↪ for method ngAfterViewInit in class CreateCourseComponent (https://goo.
   ↪ gl/w1Nwk3)
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
   ↪  9]: for (... in ...) statements must be filtered with an if statement
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
   ↪  38]: missing whitespace
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
   ↪  17]: Identifier 's' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[18,
   ↪  21]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[23,
   ↪  13]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[23,
   ↪  34]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
   ↪  12]: missing whitespace
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[6,
   ↪ 13]: The selector of the component "StatisticsComponent" should be named
   ↪  kebab-case and include dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[36, 5]: if
   ↪  statements must be braced
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[35, 17]:
   ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[38, 9]:
   ↪ Identifier 'request' is never reassigned; use 'const' instead of 'let'.
```

```
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[4, 24]: "
    ↪ should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[23, 21]: "
    ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[39, 22]: "
    ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 38]:
    ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 39]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[29, 21]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[30, 21]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[30, 35]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[30, 28]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[9, 3]: comment
    ↪ must start with a space
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[35, 21]: missing
    ↪ whitespace
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[5, 24]: " should
    ↪ be '
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[28, 21]: " should
    ↪ be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[21, 78]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[22, 46]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[28, 13]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 33]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 39]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 31]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[22, 8]:
    ↪ missing whitespace
```

```
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[10, 50]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 12]: "
    ↪ should be '
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 29]: "
    ↪ should be '
ERROR: /angular/src/app/register/register.component.ts[32, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[33, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[34, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[35, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[36, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[37, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[38, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[45, 3]: Declaration of
    ↪ instance field not allowed after declaration of instance method. Instead
    ↪ , this should come at the beginning of the class/interface.
ERROR: /angular/src/app/register/register.component.ts[36, 17]: Type string
    ↪ trivially inferred from a string literal, remove type annotation
ERROR: /angular/src/app/register/register.component.ts[37, 16]: Type string
    ↪ trivially inferred from a string literal, remove type annotation
ERROR: /angular/src/app/register/register.component.ts[25, 4]: missing
    ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[52, 18]: missing
    ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[53, 76]: missing
    ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[54, 11]: Identifier '
    ↪ data' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/register/register.component.ts[6, 29]: " should be '
ERROR: /angular/src/app/register/register.component.ts[17, 56]: " should be '
ERROR: /angular/src/app/register/register.component.ts[17, 68]: " should be '
ERROR: /angular/src/app/register/register.component.ts[17, 85]: " should be '
```

```
ERROR: /angular/src/app/register/register.component.ts[17, 97]: " should be '
ERROR: /angular/src/app/register/register.component.ts[36, 26]: " should be '
ERROR: /angular/src/app/register/register.component.ts[37, 25]: " should be '
ERROR: /angular/src/app/register/register.component.ts[46, 28]: " should be '
ERROR: /angular/src/app/register/register.component.ts[53, 35]: " should be '
ERROR: /angular/src/app/register/register.component.ts[58, 24]: " should be '
ERROR: /angular/src/app/register/register.component.ts[53, 7]: missing
   ↪ whitespace
ERROR: /angular/src/app/register/register.component.ts[41, 3]: Implement
   ↪ lifecycle hook interface OnInit for method ngOnInit in class
   ↪ RegisterComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[21, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[26, 53]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[27, 13]: Identifier 'module' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[22, 49]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[22, 50]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[22, 31]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[22, 32]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[26, 10]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[14, 3]: Implement lifecycle hook interface OnInit for method
   ↪ ngOnInit in class EditCourseComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/app.module.ts[10, 3]: comment must start with a space
ERROR: /angular/src/app/app.module.ts[14, 1]: Exceeds maximum line length of
   ↪ 140
ERROR: /angular/src/app/app.module.ts[27, 29]: " should be '
ERROR: /angular/src/app/app.module.ts[31, 23]: " should be '
ERROR: /angular/src/app/app.module.ts[49, 35]: " should be '
ERROR: /angular/src/app/app.module.ts[53, 32]: " should be '
ERROR: /angular/src/app/app.module.ts[74, 14]: " should be '
ERROR: /angular/src/app/app.module.ts[78, 13]: " should be '
ERROR: /angular/src/app/app.module.ts[82, 14]: " should be '
ERROR: /angular/src/app/app.module.ts[86, 14]: " should be '
ERROR: /angular/src/app/app.module.ts[92, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[104, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[109, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[113, 15]: " should be '
```

```
ERROR: /angular/src/app/app.module.ts[117, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[123, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[130, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[134, 19]: " should be '
ERROR: /angular/src/app/app.module.ts[142, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[147, 17]: " should be '
ERROR: /angular/src/app/app.module.ts[78, 13]: missing whitespace
ERROR: /angular/src/polyfills.ts[45, 8]: " should be '
ERROR: /angular/src/app/base-test.ts[17, 24]: Too many spaces before 'from'
ERROR: /angular/src/app/base-test.ts[4, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/base-test.ts[30, 22]: missing whitespace
ERROR: /angular/src/app/base-test.ts[32, 49]: missing whitespace
ERROR: /angular/src/app/base-test.ts[55, 22]: missing whitespace
ERROR: /angular/src/app/base-test.ts[56, 44]: missing whitespace
ERROR: /angular/src/app/base-test.ts[63, 51]: missing whitespace
ERROR: /angular/src/app/base-test.ts[65, 22]: missing whitespace
ERROR: /angular/src/app/base-test.ts[66, 44]: missing whitespace
ERROR: /angular/src/app/base-test.ts[73, 57]: missing whitespace
ERROR: /angular/src/app/base-test.ts[75, 22]: missing whitespace
ERROR: /angular/src/app/base-test.ts[76, 44]: missing whitespace
ERROR: /angular/src/app/base-test.ts[33, 9]: Identifier 'base' is never
   ↪ reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/base-test.ts[64, 9]: Identifier 'data' is never
   ↪ reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/base-test.ts[74, 9]: Identifier 'data' is never
   ↪ reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/base-test.ts[21, 29]: " should be '
ERROR: /angular/src/app/base-test.ts[55, 7]: missing whitespace
ERROR: /angular/src/app/base-test.ts[56, 10]: missing whitespace
ERROR: /angular/src/app/base-test.ts[65, 7]: missing whitespace
ERROR: /angular/src/app/base-test.ts[66, 10]: missing whitespace
ERROR: /angular/src/app/base-test.ts[75, 7]: missing whitespace
ERROR: /angular/src/app/base-test.ts[76, 10]: missing whitespace
ERROR: /angular/src/app/app.component.spec.ts[10, 7]: Identifier 'base' is
   ↪ never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/admin-page/admin-page.component.spec.ts[16, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/profiles/profiles.component.spec.ts[12, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/user-detail/user-detail.component.spec.ts[12, 9]:
   ↪  Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[13, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course/course.component.spec.ts[16, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
```

```
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.
   ↪ spec.ts[21, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
   ↪ choice.component.spec.ts[12, 9]: Identifier 'base' is never reassigned;
   ↪ use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
   ↪ .spec.ts[12, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.spec.ts[6, 24]: Too many spaces before 'from'
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.spec.ts[19, 9]: Identifier 'base' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/dashboard/dashboard.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/module/module.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
   ↪ .component.spec.ts[15, 9]: Identifier 'base' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/question/question.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/loader/loader.component.spec.ts[12, 9]: Identifier '
   ↪ base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/login/login.component.spec.ts[13, 9]: Identifier 'base'
   ↪  is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/menu/menu.component.spec.ts[12, 9]: Identifier 'base'
   ↪ is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/page-not-found/page-not-found.component.spec.ts[12, 9]:
   ↪  Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.spec.
   ↪ ts[13, 9]: Identifier 'base' is never reassigned; use 'const' instead of
   ↪  'let'.
ERROR: /angular/src/app/profile/profile-page/profile-page.component.spec.ts[12,
   ↪  9]: Identifier 'base' is never reassigned; use 'const' instead of 'let
   ↪ '.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.spec.ts[13,
   ↪ 9]: Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/register/register.component.spec.ts[12, 9]: Identifier
   ↪ 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/user.service.spec.ts[6, 24]: " should be '

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
```

```
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: ‘ng lint‘
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_16_22_428Z-debug.log
```

pep8 Output

```
learning_base/models.py:5:1: E302 expected 2 blank lines, found 1
learning_base/models.py:21:1: E302 expected 2 blank lines, found 1
learning_base/models.py:23:80: E501 line too long (88 > 79 characters)
learning_base/models.py:41:80: E501 line too long (106 > 79 characters)
learning_base/models.py:63:1: E302 expected 2 blank lines, found 1
learning_base/models.py:65:80: E501 line too long (85 > 79 characters)
learning_base/models.py:66:80: E501 line too long (84 > 79 characters)
learning_base/models.py:83:80: E501 line too long (104 > 79 characters)
learning_base/models.py:95:5: E265 block comment should start with '# '
learning_base/models.py:117:14: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/models.py:117:16: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/serializers.py:22:80: E501 line too long (96 > 79 characters)
learning_base/serializers.py:31:80: E501 line too long (92 > 79 characters)
learning_base/serializers.py:50:80: E501 line too long (95 > 79 characters)
learning_base/serializers.py:56:80: E501 line too long (82 > 79 characters)
learning_base/views.py:10:80: E501 line too long (80 > 79 characters)
learning_base/views.py:21:1: E302 expected 2 blank lines, found 1
learning_base/views.py:36:80: E501 line too long (100 > 79 characters)
learning_base/views.py:42:1: E302 expected 2 blank lines, found 1
learning_base/views.py:51:25: E231 missing whitespace after ','
learning_base/views.py:55:80: E501 line too long (100 > 79 characters)
learning_base/views.py:56:80: E501 line too long (103 > 79 characters)
learning_base/views.py:71:5: E265 block comment should start with '# '
learning_base/views.py:74:1: E302 expected 2 blank lines, found 1
learning_base/views.py:79:48: E703 statement ends with a semicolon
learning_base/views.py:81:15: E712 comparison to False should be 'if cond is
    ↪ False:' or 'if not cond:'
learning_base/views.py:88:1: E302 expected 2 blank lines, found 1
learning_base/views.py:93:1: E302 expected 2 blank lines, found 1
```

```
learning_base/views.py:96:13: E711 comparison to None should be 'if cond is
    ↪ None:'
learning_base/views.py:109:25: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:109:27: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:109:52: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:109:54: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:119:80: E501 line too long (102 > 79 characters)
learning_base/views.py:124:29: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:124:31: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:124:57: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:124:59: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:128:9: E265 block comment should start with '# '
learning_base/views.py:131:80: E501 line too long (85 > 79 characters)
learning_base/views.py:141:61: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:141:63: E251 unexpected spaces around keyword /
    ↪ parameter equals
learning_base/views.py:142:80: E501 line too long (81 > 79 characters)
learning_base/views.py:171:48: E703 statement ends with a semicolon
learning_base/views.py:173:15: E712 comparison to False should be 'if cond is
    ↪ False:' or 'if not cond:'
learning_base/views.py:194:80: E501 line too long (99 > 79 characters)
learning_base/migrations/0001_initial.py:21:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:22:80: E501 line too long (215 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:23:80: E501 line too long (149 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:24:80: E501 line too long (280 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:25:80: E501 line too long (105 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:31:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:32:80: E501 line too long (113 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:38:80: E501 line too long (114 > 79
    ↪ characters)
```

```
learning_base/migrations/0001_initial.py:39:80: E501 line too long (136 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:40:80: E501 line too long (128 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:41:80: E501 line too long (219 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:50:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:51:80: E501 line too long (101 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:52:80: E501 line too long (106 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:58:80: E501 line too long (114 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:59:80: E501 line too long (131 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:60:80: E501 line too long (134 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:61:80: E501 line too long (146 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:62:80: E501 line too long (176 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:71:80: E501 line too long (201 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:72:80: E501 line too long (93 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:82:80: E501 line too long (195 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:87:80: E501 line too long (81 > 79
    ↪ characters)
learning_base/migrations/0001_initial.py:92:80: E501 line too long (99 > 79
    ↪ characters)
learning_base/question/models.py:4:1: E302 expected 2 blank lines, found 1
learning_base/question/models.py:6:80: E501 line too long (87 > 79 characters)
learning_base/question/models.py:7:80: E501 line too long (83 > 79 characters)
learning_base/question/serializer.py:6:1: E302 expected 2 blank lines, found 1
learning_base/question/serializer.py:13:66: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:13:68: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:13:80: E501 line too long (104 > 79
    ↪ characters)
learning_base/question/serializer.py:21:1: E302 expected 2 blank lines, found 1
learning_base/question/serializer.py:28:73: E251 unexpected spaces around
    ↪ keyword / parameter equals
```

```
learning_base/question/serializer.py:28:75: E251 unexpected spaces around
    ↪ keyword / parameter equals
learning_base/question/serializer.py:28:80: E501 line too long (111 > 79
    ↪ characters)
learning_base/question/serializer.py:31:80: E501 line too long (80 > 79
    ↪ characters)
learning_base/question/multiply_choice/models.py:4:1: E302 expected 2 blank
    ↪ lines, found 1
learning_base/question/multiply_choice/models.py:41:80: E501 line too long (104
    ↪  > 79 characters)
learning_base/question/multiply_choice/models.py:56:21: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:56:23: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:57:19: E201 whitespace after
    ↪ '('
learning_base/question/multiply_choice/models.py:57:61: E202 whitespace before
    ↪ ')'
learning_base/question/multiply_choice/models.py:59:22: E225 missing whitespace
    ↪  around operator
learning_base/question/multiply_choice/models.py:67:70: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:67:72: E251 unexpected spaces
    ↪ around keyword / parameter equals
learning_base/question/multiply_choice/models.py:67:80: E501 line too long (85
    ↪ > 79 characters)
learning_base/question/multiply_choice/serializer.py:4:1: E302 expected 2 blank
    ↪  lines, found 1
learning_base/question/multiply_choice/serializer.py:18:1: E302 expected 2
    ↪ blank lines, found 1
learning_base/question/multiply_choice/serializer.py:19:51: E251 unexpected
    ↪ spaces around keyword / parameter equals
learning_base/question/multiply_choice/serializer.py:19:53: E251 unexpected
    ↪ spaces around keyword / parameter equals
```

coverage Output

Coverage report: 81%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	11	0	0	100%
django/learning_base/drag_and_drop/__init__.py	0	0	0	100%
django/learning_base/drag_and_drop/models.py	4	0	0	100%
django/learning_base/drag_and_drop/serializer.py	4	0	0	100%
django/learning_base/models.py	103	19	0	82%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	26	4	0	85%
django/learning_base/multiple_choice/serializer.py	39	1	0	97%
django/learning_base/serializers.py	139	27	0	81%
django/learning_base/tests.py	159	0	0	100%
django/learning_base/views.py	209	87	0	58%
django/manage.py	13	6	0	54%
Total	**739**	**144**	**0**	**81%**

coverage.py v4.4.1, created at 2017-09-09 15:44

Datensicherheit

bandit Output

```
Run started:2017-09-12 10:27:48.165972

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 1649
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.4 Iteration 10 - 20.07.

Ab Iteration 10 haben wir den gesammten QS-Prozess in seiner jetzigen Form durchgeführt.

Abgegebene Userstories: 17, 18

Commit Hash: f5845985bf7463a32f0f4f04279ea7481a32daaf

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 17

Reviewer: Leonhard Wiedmann
Entwickler: Ilhan Simsiki
Commithash: 6304ff4 diff a75667d
Datum: 20.07.2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- [x] Wurden nur Elemente des Material Designs verwendet? ✓
- [x] Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- [ ] Sind Buttons durch ihre Position direkt ersichtlich?
  keine Buttons
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- [x] Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- [x] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- [x] Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- [x] HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- [x] Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- [x] Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- [x] Werden alle Nutzereingaben nach Fehlern gefiltert?
- [x] Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- [x] Gibt es Meldungen der Tools?
  Nein
- [x] Gibt es obsoleten Code?
- [x] Gibt es unbenutzte Variablen?
- [x] Wurden bereits vorhandene Funktionalitäten neu implementiert?
  Nein

☒ Wurden Hilfsmethoden ausgelagert?

☒ Für das Frontend:
keine neuen Post methode

- ☐ Sind alle POST Methoden richtig formatiert?

☒ Für das Backend:

- ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
  keine neuen Schnittstelle
  - ~~☐ Wenn ja: Sind diese begründet und dokumentiert?~~
- ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  ja
- ☐ Wurde die Datenbankenstruktur geändert?
  Nein
  - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?
  Nein, Namen fehlen. wird ~~nachgerei~~ nachgearbeitet

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
    ja
  - ☐ Autor
    fehlt

3

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
    ja
  - ☐ Autor
    [illegible]
  - ☒ Eingabeparameter
  - ☒ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
  Nein
- ☐ Sind die Kommentare verständlich für alle Entwickler?
  ja
- ☐ Ist die Dokumentation im Wiki vollständig?
  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?
  ja
- ☐ Wie hoch ist die Statement Coverage?
  74%

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US - 18

Reviewer: Ilkan
Entwickler: Tobi
Commithash: 85945
Datum: 21.07

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

22.07

- ☑ Wurden nur Elemente des Material Designs verwendet? und fehlt komplett
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich? // wie besprochen ändern...
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☑ Gibt es Meldungen der Tools? Nein
- ☑ Gibt es obsoleten Code? user-detail.component.ts
  ↳ auskommentierter Array!
- ☑ Gibt es unbenutzte Variablen? ↗
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

22.07

- ☐ Wurden Hilfsmethoden ausgelagert? Keine
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert? Nein
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

22.08

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst: UserDetailUserComponent fehlt!

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

3

- [x] Sind alle Methoden kommentiert?

  Dies umfasst: alle methoden der klasse nicht vorhanden!

  - [x] Beschreibung der Funktion und Verwendung
  - [x] Autor
  - [x] Eingabeparameter
  - [x] Rückgabewert
- [x] Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- [x] Sind die Kommentare verständlich für alle Entwickler?
- [x] Ist die Dokumentation im Wiki vollständig?

Tests

- [x] Sind Tests aller Methoden vorhanden? fehlen
- [ ] Wie hoch ist die Statement Coverage?

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
npm info lifecycle clonecademy@0.0.0~lint: clonecademy@0.0.0

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/error-message/error-message.component.ts[17, 26]:
    ↪ missing whitespace
ERROR: /angular/src/app/error-message/error-message.component.ts[17, 13]: !=
    ↪ should be !==
ERROR: /angular/src/app/error-message/error-message.component.ts[17, 7]:
    ↪ missing whitespace
ERROR: /angular/src/app/error-message/error-message.component.ts[22, 3]:
    ↪ Implement lifecycle hook interface OnInit for method ngOnInit in class
    ↪ ErrorMessageComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/service/error.service.ts[12, 21]: missing whitespace
ERROR: /angular/src/app/service/error.service.ts[13, 11]: Identifier 'dialogRef
    ↪ ' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/loader/loader.component.ts[10, 42]: expected nospace
    ↪ before colon in parameter
ERROR: /angular/src/app/loader/loader.component.ts[12, 3]: Implement lifecycle
    ↪ hook interface AfterViewInit for method ngAfterViewInit in class
    ↪ LoaderComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/service/sass-helper/sass-helper.ts[20, 13]: Identifier
    ↪ 'bodyStyles' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/sass-helper/sass-helper.ts[12, 37]: missing
    ↪ whitespace
ERROR: /angular/src/app/service/sass-helper/sass-helper.ts[6, 15]: The selector
    ↪  of the component "SassHelperComponent" should have prefix "app" (https
    ↪ ://goo.gl/cix8BY)
ERROR: /angular/src/app/app.component.ts[14, 78]: missing whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[45, 9]: comment
    ↪ must start with a space
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[28, 32]: missing
    ↪  whitespace
```

```
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[42, 40]: missing
    ↪  whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[44, 20]: missing
    ↪  whitespace
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[43, 9]:
    ↪ Identifier 'percent' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[15, 30]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[15, 40]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[16, 30]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[16, 40]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[48, 12]: "
    ↪ should be '
ERROR: /angular/src/app/learn/dashboard/dashboard.component.ts[44, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[2, 26]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[3, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[15, 34]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[16, 29]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/menu/menu.component.ts[16, 72]: missing whitespace
ERROR: /angular/src/app/menu/menu.component.ts[17, 34]: " should be '
ERROR: /angular/src/app/directive/module.directive.ts[10, 29]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[10, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[24, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[28, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[31, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[2, 31]: " should be '
ERROR: /angular/src/app/learn/question/question.module.ts[25, 12]: " should be
    ↪ '
ERROR: /angular/src/app/learn/question/question.module.ts[7, 13]: The selector
    ↪ of the component "QuestionModule" should be named kebab-case and include
    ↪  dash (https://goo.gl/mBg67Z)
```

```
ERROR: /angular/src/app/learn/question/question.module.ts[10, 14]: The name of
    ↪ the class QuestionModule should end with the suffix Component (https://
    ↪ goo.gl/5X1TE7)
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[24, 7]: comment must start with a space
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[17, 16]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 20]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[19, 14]: Identifier 'ans' is never reassigned; use 'const'
    ↪  instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[18, 9]: Identifier 'sendAnswer' is never reassigned; use '
    ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[3, 32]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[4, 31]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[13, 29]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 9]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[7, 13]: The selector of the component "
    ↪ MultipleChoiceQuestionComponent" should be named kebab-case and include
    ↪ dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/module/module.component.ts[40, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[9, 35]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[10, 49]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[20, 5]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 21]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 46]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 49]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 50]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 45]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 31]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 32]: missing
    ↪ whitespace
```

```
ERROR: /angular/src/app/learn/module/module.component.ts[28, 14]: missing
    ↪ whitespace
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
    ↪  9]: for (... in ...) statements must be filtered with an if statement
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
    ↪  38]: missing whitespace
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
    ↪  17]: Identifier 's' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[18,
    ↪  21]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[23,
    ↪  13]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[23,
    ↪  34]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
    ↪  12]: missing whitespace
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[6,
    ↪ 13]: The selector of the component "StatisticsComponent" should be named
    ↪  kebab-case and include dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[36, 5]: if
    ↪  statements must be braced
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[35, 17]:
    ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[38, 9]:
    ↪ Identifier 'request' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[4, 24]: "
    ↪ should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[23, 21]: "
    ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[39, 22]: "
    ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 38]:
    ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 39]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[37, 21]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[47, 12]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[38, 21]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[38, 35]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[42, 37]: "
    ↪ should be '
```

```
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[42, 55]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[48, 22]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[48, 41]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[42, 24]: !=
    ↪ should be !==
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[38, 28]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[48, 29]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[33, 3]:
    ↪ Implement lifecycle hook interface OnInit for method ngOnInit in class
    ↪ UserDetailComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[9, 3]: comment
    ↪ must start with a space
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[35, 21]: missing
    ↪ whitespace
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[5, 24]: " should
    ↪ be '
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[28, 21]: " should
    ↪ be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[21, 78]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[28, 13]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 33]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 39]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 31]:
    ↪  " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [30, 18]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 80]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [32, 15]: Identifier 'data' is never reassigned; use 'const' instead of
    ↪ 'let'.
```

```
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [6, 29]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 56]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 68]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 85]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 97]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 39]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [33, 28]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [35, 35]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 11]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [27, 5]: Implement lifecycle hook interface OnInit for method ngOnInit
    ↪ in class UserDetailUserComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[10, 50]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 12]: "
    ↪ should be '
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 29]: "
    ↪ should be '
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[14, 16]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[25, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[33, 54]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[41, 10]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[43, 16]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[34, 13]: Identifier 'module' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[44, 11]: Identifier 'course' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[42, 9]: Identifier 'saveModules' is never reassigned; use 'const'
    ↪ instead of 'let'.
```

```
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[48, 27]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[27, 49]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[27, 50]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[27, 31]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[27, 32]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[33, 10]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[43, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[14, 3]: Implement lifecycle hook interface OnChanges for method
    ↪ ngOnChanges in class EditCourseComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[18, 3]: Implement lifecycle hook interface OnInit for method
    ↪ ngOnInit in class EditCourseComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/app.module.ts[10, 3]: comment must start with a space
ERROR: /angular/src/app/app.module.ts[14, 1]: Exceeds maximum line length of
    ↪ 140
ERROR: /angular/src/app/app.module.ts[27, 29]: " should be '
ERROR: /angular/src/app/app.module.ts[31, 23]: " should be '
ERROR: /angular/src/app/app.module.ts[49, 35]: " should be '
ERROR: /angular/src/app/app.module.ts[53, 32]: " should be '
ERROR: /angular/src/app/app.module.ts[75, 14]: " should be '
ERROR: /angular/src/app/app.module.ts[79, 13]: " should be '
ERROR: /angular/src/app/app.module.ts[83, 14]: " should be '
ERROR: /angular/src/app/app.module.ts[87, 14]: " should be '
ERROR: /angular/src/app/app.module.ts[93, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[105, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[110, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[114, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[118, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[124, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[131, 15]: " should be '
ERROR: /angular/src/app/app.module.ts[135, 19]: " should be '
ERROR: /angular/src/app/app.module.ts[143, 11]: " should be '
ERROR: /angular/src/app/app.module.ts[148, 17]: " should be '
ERROR: /angular/src/app/app.module.ts[79, 13]: missing whitespace
ERROR: /angular/src/polyfills.ts[45, 8]: " should be '
ERROR: /angular/src/app/base-test.ts[17, 24]: Too many spaces before 'from'
ERROR: /angular/src/app/base-test.ts[4, 1]: Exceeds maximum line length of 140
ERROR: /angular/src/app/base-test.ts[30, 22]: missing whitespace
```

```
ERROR: /angular/src/app/base-test.ts[32, 49]: missing whitespace
ERROR: /angular/src/app/base-test.ts[55, 22]: missing whitespace
ERROR: /angular/src/app/base-test.ts[56, 44]: missing whitespace
ERROR: /angular/src/app/base-test.ts[63, 51]: missing whitespace
ERROR: /angular/src/app/base-test.ts[65, 22]: missing whitespace
ERROR: /angular/src/app/base-test.ts[66, 44]: missing whitespace
ERROR: /angular/src/app/base-test.ts[73, 57]: missing whitespace
ERROR: /angular/src/app/base-test.ts[75, 22]: missing whitespace
ERROR: /angular/src/app/base-test.ts[76, 44]: missing whitespace
ERROR: /angular/src/app/base-test.ts[33, 9]: Identifier 'base' is never
   ↪ reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/base-test.ts[64, 9]: Identifier 'data' is never
   ↪ reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/base-test.ts[74, 9]: Identifier 'data' is never
   ↪ reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/base-test.ts[21, 29]: " should be '
ERROR: /angular/src/app/base-test.ts[55, 7]: missing whitespace
ERROR: /angular/src/app/base-test.ts[56, 10]: missing whitespace
ERROR: /angular/src/app/base-test.ts[65, 7]: missing whitespace
ERROR: /angular/src/app/base-test.ts[66, 10]: missing whitespace
ERROR: /angular/src/app/base-test.ts[75, 7]: missing whitespace
ERROR: /angular/src/app/base-test.ts[76, 10]: missing whitespace
ERROR: /angular/src/app/server.service.ts[14, 24]: missing whitespace
ERROR: /angular/src/app/test/test.component.ts[16, 12]: missing whitespace
ERROR: /angular/src/app/app.component.spec.ts[10, 7]: Identifier 'base' is
   ↪ never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/admin-page/admin-page.component.spec.ts[16, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/profiles/profiles.component.spec.ts[12, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/user-detail/user-detail.component.spec.ts[16, 9]:
   ↪  Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[28, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[44, 13]:
   ↪ Identifier 'dialogRef' is never reassigned; use 'const' instead of 'let
   ↪ '.
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[25, 7]:
   ↪ Identifier 'fixture' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course/course.component.spec.ts[16, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.
   ↪ spec.ts[21, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
```

```
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
   ↪ choice.component.spec.ts[12, 9]: Identifier 'base' is never reassigned;
   ↪ use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
   ↪ .spec.ts[12, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.spec.ts[6, 24]: Too many spaces before 'from'
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.spec.ts[19, 9]: Identifier 'base' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/dashboard/dashboard.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/module/module.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
   ↪ .component.spec.ts[15, 9]: Identifier 'base' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/question/question.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/loader/loader.component.spec.ts[12, 9]: Identifier '
   ↪ base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/login/login.component.spec.ts[13, 9]: Identifier 'base'
   ↪  is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/menu/menu.component.spec.ts[12, 9]: Identifier 'base'
   ↪ is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/page-not-found/page-not-found.component.spec.ts[12, 9]:
   ↪  Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.spec.
   ↪ ts[13, 9]: Identifier 'base' is never reassigned; use 'const' instead of
   ↪  'let'.
ERROR: /angular/src/app/profile/profile-page/profile-page.component.spec.ts[12,
   ↪  9]: Identifier 'base' is never reassigned; use 'const' instead of 'let
   ↪ '.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.spec.ts[13,
   ↪ 9]: Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.
   ↪ spec.ts[12, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/register/register.component.spec.ts[12, 9]: Identifier
   ↪ 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/user.service.spec.ts[6, 24]: " should be '

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
```

```
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: ‘ng lint‘
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_17_00_435Z-debug.log
```

pylint Output

```
************* Module learning_base.models
W: 48, 0: TODO: Implement correct user profile access string (fixme)
W: 59, 0: TODO: Refactor these to a decorator (fixme)
E: 36,16: Module 'django.utils.timezone' has no 'today' member (no-member)
E: 56,13: Module 'django.utils.timezone' has no 'localdate' member (no-member)
R:254, 4: Method could be a function (no-self-use)
W:281, 0: Found __unicode__ method on model (Try). Python3 uses __str__. (model
    ↪ -has-unicode)
E:320, 4: Method should have "self" as first argument (no-self-argument)
E:321,18: Module 'django.db.models' has no 'Course' member (no-member)
W: 2, 0: Unused apps imported from django.apps (unused-import)
************* Module learning_base.views
W:292, 0: TODO: probably should be check_permissions(self, request) (fixme)
W:352, 0: TODO Implement saving a users data (fixme)
W:413, 0: TODO: implement proper send_mail() (fixme)
W:445, 0: TODO: fix if an localization issues arrise (fixme)
W:484, 0: TODO Find out if it is usefull to send a 200 when a user was already
    ↪ mod (fixme)
C: 67,12: Invalid variable name "TYPES" (invalid-name)
C: 68,12: Invalid variable name "CATEGORIES" (invalid-name)
W: 68,29: Lambda may not be necessary (unnecessary-lambda)
C: 69,12: Invalid variable name "LANGUAGES" (invalid-name)
C: 93, 8: Invalid variable name "e" (invalid-name)
C:120, 8: Invalid variable name "e" (invalid-name)
C:150, 8: Invalid variable name "e" (invalid-name)
C:165, 8: Invalid variable name "id" (invalid-name)
R:210, 8: Consider using ternary (question.module.is_first_module if question.
    ↪ is_first_question else request.user.try_set.filter(question__order=(
    ↪ question.order) - (1), solved=True).exists()) (consider-using-ternary)
R:209, 4: Method could be a function (no-self-use)
C:232, 8: Invalid variable name "e" (invalid-name)
C:245, 8: Invalid variable name "e" (invalid-name)
```

```
E:446,34: Module 'django.utils.timezone' has no 'localdate' member (no-member)
C: 14, 0: Imports from package rest_framework are not grouped (ungrouped-
    ↪ imports)
C: 15, 0: Imports from package django are not grouped (ungrouped-imports)
************* Module learning_base.serializers
W:250, 0: TODO add language to profile (fixme)
W: 3, 0: Wildcard import models (wildcard-import)
W: 4, 0: Wildcard import learning_base.multiple_choice.models (wildcard-import)
W: 5, 0: Wildcard import learning_base.drag_and_drop.models (wildcard-import)
W: 6, 0: Wildcard import learning_base.multiple_choice.serializer (wildcard-
    ↪ import)
W: 7, 0: Wildcard import learning_base.drag_and_drop.serializer (wildcard-
    ↪ import)
W: 10, 0: Method 'create' is abstract in class 'BaseSerializer' but is not
    ↪ overridden (abstract-method)
W: 10, 0: Method 'to_internal_value' is abstract in class 'BaseSerializer' but
    ↪ is not overridden (abstract-method)
W: 10, 0: Method 'update' is abstract in class 'BaseSerializer' but is not
    ↪ overridden (abstract-method)
C:186, 8: Invalid variable name "e" (invalid-name)
E:188,36: Instance of 'Exception' has no 'detail' member (no-member)
C:228, 8: Invalid variable name "p" (invalid-name)
W:289, 0: Method 'create' is abstract in class 'BaseSerializer' but is not
    ↪ overridden (abstract-method)
W:289, 0: Method 'to_internal_value' is abstract in class 'BaseSerializer' but
    ↪ is not overridden (abstract-method)
W:289, 0: Method 'update' is abstract in class 'BaseSerializer' but is not
    ↪ overridden (abstract-method)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import apps from wildcard import (unused-wildcard-import)
W: 4, 0: Unused import MultipleChoiceAnswer from wildcard import (unused-
    ↪ wildcard-import)
W: 4, 0: Unused import MultipleChoiceQuestion from wildcard import (unused-
    ↪ wildcard-import)
W: 5, 0: Unused import DragAndDropQuestion from wildcard import (unused-
    ↪ wildcard-import)
W: 6, 0: Unused import MultipleChoiceQuestionEditSerializer from wildcard
    ↪ import (unused-wildcard-import)
W: 6, 0: Unused import MultipleChoiceAnswerEditSerializer from wildcard import
    ↪ (unused-wildcard-import)
```

```
W: 6, 0: Unused import MultipleChoiceQuestionPreviewSerializer from wildcard
    ↪ import (unused-wildcard-import)
W: 6, 0: Unused import MultipleChoiceAnswerSerializer from wildcard import (
    ↪ unused-wildcard-import)
W: 7, 0: Unused import DragAndDropSerializer from wildcard import (unused-
    ↪ wildcard-import)
C: 4, 0: external import "from learning_base.multiple_choice.models import *"
    ↪ should be placed before "from .models import *" (wrong-import-order)
C: 5, 0: external import "from learning_base.drag_and_drop.models import *"
    ↪ should be placed before "from .models import *" (wrong-import-order)
C: 6, 0: external import "from learning_base.multiple_choice.serializer import
    ↪ *" should be placed before "from .models import *" (wrong-import-order)
C: 7, 0: external import "from learning_base.drag_and_drop.serializer import *"
    ↪  should be placed before "from .models import *" (wrong-import-order)
************* Module learning_base.admin
W: 3, 0: Wildcard import learning_base.models (wildcard-import)
W: 4, 0: Wildcard import learning_base.multiple_choice.models (wildcard-import)
W: 3, 0: Unused import User from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import apps from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
************* Module learning_base.multiple_choice.serializer
W: 3, 0: Wildcard import models (wildcard-import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
************* Module learning_base.multiple_choice.models
R: 10, 4: Method could be a function (no-self-use)
R: 34, 4: Method could be a function (no-self-use)
R: 38, 4: Method could be a function (no-self-use)
R: 64, 4: Method could be a function (no-self-use)
R: 68, 4: Method could be a function (no-self-use)
************* Module learning_base.drag_and_drop.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import DragAndDropQuestion from wildcard import (unused-
    ↪ wildcard-import)
************* Module learning_base.drag_and_drop.models
W: 1, 0: Unused models imported from django.db (unused-import)

------------------------------------------------------------------
```

```
Your code has been rated at 9.53/10 (previous run: 6.71/10, +2.82)
```

coverage Output

Coverage report: 74%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	11	0	0	100%
django/learning_base/drag_and_drop/__init__.py	0	0	0	100%
django/learning_base/drag_and_drop/models.py	4	0	0	100%
django/learning_base/drag_and_drop/serializer.py	4	0	0	100%
django/learning_base/models.py	104	19	0	82%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	33	10	0	70%
django/learning_base/multiple_choice/serializer.py	52	21	0	60%
django/learning_base/serializers.py	192	60	0	69%
django/learning_base/tests.py	159	2	0	99%
django/learning_base/views.py	225	98	0	56%
django/manage.py	13	6	0	54%
Total	**829**	**216**	**0**	**74%**

coverage.py v4.4.1, created at 2017-09-09 15:46

Datensicherheit

bandit Output

```
Run started:2017-09-12 10:27:57.616950

Test results:
>> Issue: [B104:hardcoded_bind_all_interfaces] Possible binding to all
   ↪ interfaces.
   Severity: Medium Confidence: Medium
   Location: django/composeexample/settings.py:28
27
28 ALLOWED_HOSTS = ['0.0.0.0']
29

--------------------------------------------------

Code scanned:
       Total lines of code: 2151
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 1.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 1.0
              High: 0.0
Files skipped (0):
```

C.5 Iteration 11 - 27.07.

Abgegebene Userstories: 22

Commit Hash: 2d17b38eecc8bbd754534a564ff2a22d60ee5f71

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US - 22

Reviewer: ilhan
Entwickler: Leon
Commithash: 2d17b
Datum: 28.07

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☑ Gibt es Meldungen der Tools? Nein
- ☑ Gibt es obsoleten Code? Nein
- ☑ Gibt es unbenutzte Variablen? Nein
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?

    Neuer Fragentyp
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst: Image Cropper Dialog, Add Multiply Choice

  Fehlen!
  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

29.07

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst: openImageDialog() constructor fileChangeListener()

  fehlen!

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert
- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

  Keine

  ← auf morgen verschoben ...

Tests

- ☐ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Veränderbarkeit

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
[..................] / : info lifecycle clonecademy@0.0.0~prelint: clonecademy@

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[2, 26]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[3, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[15, 34]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[16, 29]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/menu/menu.component.ts[16, 72]: missing whitespace
ERROR: /angular/src/app/menu/menu.component.ts[17, 34]: " should be '
ERROR: /angular/src/app/directive/module.directive.ts[10, 29]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[10, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[25, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[29, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[32, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[2, 31]: " should be '
ERROR: /angular/src/app/learn/question/question.module.ts[26, 12]: " should be
    ↪ '
ERROR: /angular/src/app/learn/question/question.module.ts[7, 13]: The selector
    ↪ of the component "QuestionModule" should be named kebab-case and include
    ↪  dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/question/question.module.ts[10, 14]: The name of
    ↪ the class QuestionModule should end with the suffix Component (https://
    ↪ goo.gl/5X1TE7)
```

```
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[24, 7]: comment must start with a space
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[17, 16]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 20]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[19, 14]: Identifier 'ans' is never reassigned; use 'const'
    ↪  instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[18, 9]: Identifier 'sendAnswer' is never reassigned; use '
    ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[3, 32]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[4, 31]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[13, 29]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 9]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[7, 13]: The selector of the component "
    ↪ MultipleChoiceQuestionComponent" should be named kebab-case and include
    ↪ dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/module/module.component.ts[40, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[9, 35]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[10, 49]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[20, 5]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 21]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 46]: " should be '
ERROR: /angular/src/app/learn/module/module.component.ts[33, 49]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 50]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 45]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 31]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[33, 32]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/module/module.component.ts[28, 14]: missing
    ↪ whitespace
```

```
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
    ↪  9]: for (... in ...) statements must be filtered with an if statement
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
    ↪  38]: missing whitespace
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
    ↪  17]: Identifier 's' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[18,
    ↪  21]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[23,
    ↪  13]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[23,
    ↪  34]: " should be '
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[22,
    ↪  12]: missing whitespace
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.ts[6,
    ↪ 13]: The selector of the component "StatisticsComponent" should be named
    ↪  kebab-case and include dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[36, 5]: if
    ↪  statements must be braced
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[35, 17]:
    ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[38, 9]:
    ↪ Identifier 'request' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[4, 24]: "
    ↪ should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[23, 21]: "
    ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[39, 22]: "
    ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 38]:
    ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 39]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[37, 21]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[47, 12]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[38, 21]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[38, 35]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[42, 37]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[42, 55]: "
    ↪ should be '
```

```
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[48, 22]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[48, 41]: "
    ↪ should be '
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[42, 24]: !=
    ↪ should be !==
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[38, 28]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[48, 29]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/user-detail/user-detail.component.ts[33, 3]:
    ↪ Implement lifecycle hook interface OnInit for method ngOnInit in class
    ↪ UserDetailComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[9, 3]: comment
    ↪ must start with a space
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[35, 21]: missing
    ↪ whitespace
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[5, 24]: " should
    ↪ be '
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[28, 21]: " should
    ↪ be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[21, 78]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[28, 13]:
    ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 33]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 39]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 12]:
    ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 31]:
    ↪  " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [30, 16]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 78]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [32, 13]: Identifier 'data' is never reassigned; use 'const' instead of
    ↪ 'let'.
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [6, 29]: " should be '
```

```
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 56]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 68]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 85]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 97]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 37]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 9]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [27, 5]: Implement lifecycle hook interface OnInit for method ngOnInit
    ↪ in class UserDetailUserComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[10, 50]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 12]: "
    ↪ should be '
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 29]: "
    ↪ should be '
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[14, 16]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[25, 9]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[33, 54]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[41, 10]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[43, 16]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[34, 13]: Identifier 'module' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[44, 11]: Identifier 'course' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[42, 9]: Identifier 'saveModules' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[48, 27]: " should be '
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[27, 49]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[27, 50]: missing whitespace
```

```
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[27, 31]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[27, 32]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[33, 10]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[43, 7]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[14, 3]: Implement lifecycle hook interface OnChanges for method
   ↪ ngOnChanges in class EditCourseComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
   ↪ .ts[18, 3]: Implement lifecycle hook interface OnInit for method
   ↪ ngOnInit in class EditCourseComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/test/test.component.ts[16, 12]: missing whitespace
ERROR: /angular/src/app/app.component.spec.ts[10, 7]: Identifier 'base' is
   ↪ never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/admin-page/admin-page.component.spec.ts[16, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/profiles/profiles.component.spec.ts[12, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/user-detail/user-detail.component.spec.ts[16, 9]:
   ↪  Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[28, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[44, 13]:
   ↪ Identifier 'dialogRef' is never reassigned; use 'const' instead of 'let
   ↪ '.
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[25, 7]:
   ↪ Identifier 'fixture' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course/course.component.spec.ts[16, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.
   ↪ spec.ts[21, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
   ↪ choice.component.spec.ts[12, 9]: Identifier 'base' is never reassigned;
   ↪ use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
   ↪ .spec.ts[12, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.spec.ts[6, 24]: Too many spaces before 'from'
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.spec.ts[19, 9]: Identifier 'base' is never reassigned; use '
   ↪ const' instead of 'let'.
```

```
ERROR: /angular/src/app/learn/dashboard/dashboard.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/module/module.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
   ↪ .component.spec.ts[15, 9]: Identifier 'base' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/question/question.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/loader/loader.component.spec.ts[12, 9]: Identifier '
   ↪ base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/login/login.component.spec.ts[13, 9]: Identifier 'base'
   ↪  is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/menu/menu.component.spec.ts[12, 9]: Identifier 'base'
   ↪ is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/page-not-found/page-not-found.component.spec.ts[12, 9]:
   ↪  Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.spec.
   ↪ ts[13, 9]: Identifier 'base' is never reassigned; use 'const' instead of
   ↪  'let'.
ERROR: /angular/src/app/profile/profile-page/profile-page.component.spec.ts[12,
   ↪  9]: Identifier 'base' is never reassigned; use 'const' instead of 'let
   ↪ '.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.spec.ts[13,
   ↪ 9]: Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.
   ↪ spec.ts[12, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/register/register.component.spec.ts[12, 9]: Identifier
   ↪ 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/user.service.spec.ts[6, 24]: " should be '

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: ‘ng lint‘
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
   ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_17_30_952Z-debug.log
```

pylint Output

```
************* Module learning_base.models
W: 48, 0: TODO: Implement correct user profile access string (fixme)
W: 59, 0: TODO: Refactor these to a decorator (fixme)
E: 36,16: Module 'django.utils.timezone' has no 'today' member (no-member)
E: 56,13: Module 'django.utils.timezone' has no 'localdate' member (no-member)
E:324,18: Module 'django.db.models' has no 'Course' member (no-member)
W: 2, 0: Unused apps imported from django.apps (unused-import)
************* Module learning_base.views
W:293, 0: TODO: probably should be check_permissions(self, request) (fixme)
W:353, 0: TODO Implement saving a users data (fixme)
W:414, 0: TODO: implement proper send_mail() (fixme)
W:446, 0: TODO: fix if an localization issues arrise (fixme)
W:485, 0: TODO Find out if it is usefull to send a 200 when a user was already
    ↪ mod (fixme)
C: 67,12: Invalid variable name "TYPES" (invalid-name)
C: 68,12: Invalid variable name "CATEGORIES" (invalid-name)
C: 69,12: Invalid variable name "LANGUAGES" (invalid-name)
C:150, 8: Invalid variable name "e" (invalid-name)
C:165, 8: Invalid variable name "id" (invalid-name)
R:210, 8: Consider using ternary (question.module.is_first_module if question.
    ↪ is_first_question else request.user.try_set.filter(question__order=(
    ↪ question.order) - (1), solved=True).exists()) (consider-using-ternary)
C:245, 8: Invalid variable name "e" (invalid-name)
E:447,34: Module 'django.utils.timezone' has no 'localdate' member (no-member)
W: 1, 0: Unused render imported from django.shortcuts (unused-import)
W: 11, 0: Unused Module imported from learning_base.models (unused-import)
W: 11, 0: Unused Question imported from learning_base.models (unused-import)
C: 15, 0: Imports from package django are not grouped (ungrouped-imports)
************* Module learning_base.serializers
W:250, 0: TODO add language to profile (fixme)
W: 3, 0: Wildcard import models (wildcard-import)
W: 5, 0: Wildcard import learning_base.drag_and_drop.models (wildcard-import)
W: 7, 0: Wildcard import learning_base.drag_and_drop.serializer (wildcard-
    ↪ import)
C:186, 8: Invalid variable name "e" (invalid-name)
E:188,36: Instance of 'Exception' has no 'detail' member (no-member)
C:228, 8: Invalid variable name "p" (invalid-name)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import apps from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
```

```
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 5, 0: Unused import DragAndDropQuestion from wildcard import (unused-
    ↪ wildcard-import)
W: 6, 0: Unused import b64decode from wildcard import (unused-wildcard-import)
W: 7, 0: Unused import DragAndDropSerializer from wildcard import (unused-
    ↪ wildcard-import)
C: 5, 0: external import "from learning_base.drag_and_drop.models import *"
    ↪ should be placed before "from .models import *" (wrong-import-order)
C: 7, 0: external import "from learning_base.drag_and_drop.serializer import *"
    ↪  should be placed before "from .models import *" (wrong-import-order)
************* Module learning_base.admin
W: 3, 0: Wildcard import learning_base.models (wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import apps from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import User from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
************* Module learning_base.drag_and_drop.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import DragAndDropQuestion from wildcard import (unused-
    ↪ wildcard-import)
************* Module learning_base.drag_and_drop.models
W: 1, 0: Unused models imported from django.db (unused-import)

------------------------------------------------------------------
Your code has been rated at 9.59 (previous run: 7.00/10, +2.59)
```

coverage Output

Coverage report: 74%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	11	0	0	100%
django/learning_base/drag_and_drop/__init__.py	0	0	0	100%
django/learning_base/drag_and_drop/models.py	4	0	0	100%
django/learning_base/drag_and_drop/serializer.py	4	0	0	100%
django/learning_base/models.py	106	20	0	81%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	37	11	0	70%
django/learning_base/multiple_choice/serializer.py	53	21	0	60%
django/learning_base/serializers.py	192	60	0	69%
django/learning_base/tests.py	159	0	0	100%
django/learning_base/views.py	226	99	0	56%
django/manage.py	13	6	0	54%
Total	**837**	**217**	**0**	**74%**

coverage.py v4.4.1, created at 2017-09-09 15:48

Datensicherheit

bandit Output

```
Run started:2017-09-12 10:28:07.436143

Test results:
>> Issue: [B104:hardcoded_bind_all_interfaces] Possible binding to all
   ↪ interfaces.
   Severity: Medium Confidence: Medium
   Location: django/composeexample/settings.py:28
27
28 ALLOWED_HOSTS = ['0.0.0.0']
29

--------------------------------------------------

Code scanned:
       Total lines of code: 2237
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 1.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 1.0
              High: 0.0
Files skipped (0):
```

C.6 Iteration 12 - 03.08.

Abgegebene Userstories: 4,19,20

Commit Hash: 7af0e8284c862f9841d8101c6f4569adb0c6289c

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 4

Reviewer: Claas
Entwickler: Leon
Commithash: ~~1d5330 diff~~ c7c66ff diff b58ada a
Datum: 03.08.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?

  ja

- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?

  ja

- ☐ Sind Buttons durch ihre Position direkt ersichtlich?

  ja

- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?

  ja

- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?

  ja

- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

  Ja

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar? ja (wird unübersichtlich, schöner machen ist neue US)

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  ja

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

  ja

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

  ja

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  ja

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  ja

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  ja (wurden direkt erledigt)

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  ja (wurden gelöscht)

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  na

- ☐ Für das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?

    ja

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      Änderungen waren sinnvoll und wurden abgesprochen/dokumentiert

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

    ja

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      /

Kommentare

- ☐ Sind alle Klassen kommentiert?

  nein (Front end fehlt) wurde direkt behoben

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    /

  - ☐ Autor

    /

- ☐ Sind alle Methoden kommentiert?

  nein (Frontend fehlt, wurde direkt behoben)

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert
- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nein
- ☐ Sind die Kommentare verständlich für alle Entwickler?

  ja
- ☐ Ist die Dokumentation im Wiki vollständig?

  ja (siehe oben

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  ja
- ☐ Wie hoch ist die Statement Coverage?

  74 % (muss zum nächsten Deploy angehoben werden)

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 20

Reviewer: Leonhard Wiedmann
Entwickler: Tobias Huber
Commithash: 7ef0e828 diff 2d17b38ee
Datum: 03.08.2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?
  ja
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
  ja
- ☐ Sind Buttons durch ihre Position direkt ersichtlich?
  ja
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
  ja
- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
  ja
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?
  ja

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

  ja

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  ja

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

  ja

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

  ja

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  ja

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  ja

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  Nein

- ☐ Gibt es obsoleten Code?

  Nein

- ☐ Gibt es unbenutzte Variablen?

  Nein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  Nein

- ☐ Wurden Hilfsmethoden ausgelagert?
  Nein, keine Hilfsmethoden benötigt
- ☐ Für das Frontend:
  keine neuen Post Methoden
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  keine neuen Schnittstellen nur Erweiterung
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
      ja
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
    ja
  - ☐ Wurde die Datenbankenstruktur geändert?
    Nein
    - ☐ ~~Wenn ja: Sind diese begründet~~ und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?
  ja

  Dies umfasst:
  - ☐ Beschreibung der Funktion und Verwendung
    ja
  - ☐ Autor
    ja

- ☐ Sind alle Methoden kommentiert?

  ja

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ja

  - ☐ Autor

    ja

  - ☐ Eingabeparameter

    /

  - ☐ Rückgabewert

    /

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  Nein

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollständig?

  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  ja

- ☐ Wie hoch ist die Statement Coverage?

  74%

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
[..................] - : info using node@v8.0.0

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[2, 26]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[3, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[15, 34]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[16, 29]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/menu/menu.component.ts[16, 72]: missing whitespace
ERROR: /angular/src/app/menu/menu.component.ts[17, 34]: " should be '
ERROR: /angular/src/app/learn/question/question.module.ts[10, 46]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[25, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[29, 16]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[32, 13]: missing
    ↪ whitespace
ERROR: /angular/src/app/learn/question/question.module.ts[2, 31]: " should be '
ERROR: /angular/src/app/learn/question/question.module.ts[26, 12]: " should be
    ↪ '
ERROR: /angular/src/app/learn/question/question.module.ts[7, 13]: The selector
    ↪ of the component "QuestionModule" should be named kebab-case and include
    ↪  dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/question/question.module.ts[10, 14]: The name of
    ↪ the class QuestionModule should end with the suffix Component (https://
    ↪ goo.gl/5X1TE7)
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[24, 7]: comment must start with a space
```

```
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[17, 16]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 20]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[19, 14]: Identifier 'ans' is never reassigned; use 'const'
    ↪  instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[18, 9]: Identifier 'sendAnswer' is never reassigned; use '
    ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[3, 32]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[4, 31]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[13, 29]: " should be '
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[20, 9]: missing whitespace
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
    ↪ .component.ts[7, 13]: The selector of the component "
    ↪ MultipleChoiceQuestionComponent" should be named kebab-case and include
    ↪ dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
    ↪ ts[14, 9]: Type string trivially inferred from a string literal, remove
    ↪ type annotation
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
    ↪ ts[17, 26]: missing whitespace
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
    ↪ ts[14, 18]: " should be '
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
    ↪ ts[17, 13]: != should be !==
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
    ↪ ts[17, 7]: missing whitespace
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
    ↪ ts[8, 13]: The selector of the component "WrongFeedbackComponent" should
    ↪  have prefix "app" (https://goo.gl/cix8BY)
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
    ↪ ts[25, 3]: Implement lifecycle hook interface OnInit for method ngOnInit
    ↪  in class WrongFeedbackComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/learn/question/correct-feedback/correct-feedback.
    ↪ component.ts[18, 26]: missing whitespace
ERROR: /angular/src/app/learn/question/correct-feedback/correct-feedback.
    ↪ component.ts[18, 13]: != should be !==
ERROR: /angular/src/app/learn/question/correct-feedback/correct-feedback.
    ↪ component.ts[18, 7]: missing whitespace
```

```
ERROR: /angular/src/app/learn/question/correct-feedback/correct-feedback.
   ↪ component.ts[8, 13]: The selector of the component "
   ↪ CorrectFeedbackComponent" should have prefix "app" (https://goo.gl/
   ↪ cix8BY)
ERROR: /angular/src/app/learn/question/correct-feedback/correct-feedback.
   ↪ component.ts[24, 3]: Implement lifecycle hook interface OnInit for
   ↪ method ngOnInit in class CorrectFeedbackComponent (https://goo.gl/w1Nwk3
   ↪ )
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[36, 5]: if
   ↪  statements must be braced
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[35, 17]:
   ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[38, 9]:
   ↪ Identifier 'request' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[4, 24]: "
   ↪ should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[23, 21]: "
   ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[39, 22]: "
   ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 38]:
   ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 39]:
   ↪ missing whitespace
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[9, 3]: comment
   ↪ must start with a space
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[35, 21]: missing
   ↪ whitespace
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[5, 24]: " should
   ↪ be '
ERROR: /angular/src/app/admin/profiles/profiles.component.ts[28, 21]: " should
   ↪ be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[21, 78]:
   ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[28, 13]:
   ↪  missing whitespace
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 12]:
   ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[16, 33]:
   ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 12]:
   ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[17, 39]:
   ↪  " should be '
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 12]:
   ↪  " should be '
```

```
ERROR: /angular/src/app/profile/profile-page/profile-page.component.ts[18, 31]:
    ↪  " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [30, 16]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 78]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [32, 13]: Identifier 'data' is never reassigned; use 'const' instead of
    ↪ 'let'.
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [6, 29]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 56]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 68]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 85]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 97]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 37]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 9]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [27, 5]: Implement lifecycle hook interface OnInit for method ngOnInit
    ↪ in class UserDetailUserComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[10, 50]:
    ↪ missing whitespace
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 12]: "
    ↪ should be '
ERROR: /angular/src/app/admin/admin-page/admin-page.component.ts[13, 29]: "
    ↪ should be 'S
ERROR: /angular/src/app/learn/course-editor/create-course/edit-course.component
    ↪ .ts[18, 3]: Implement lifecycle hook interface OnInit for method
    ↪ ngOnInit in class EditCourseComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/server.service.ts[14, 24]: missing whitespace
ERROR: /angular/src/app/test/test.component.ts[16, 12]: missing whitespace
ERROR: /angular/src/app/app.component.spec.ts[10, 7]: Identifier 'base' is
    ↪ never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/admin-page/admin-page.component.spec.ts[16, 9]:
    ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/profiles/profiles.component.spec.ts[12, 9]:
    ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/admin/user-detail/user-detail.component.spec.ts[16, 9]:
    ↪  Identifier 'base' is never reassigned; use 'const' instead of 'let'.
```

```
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[28, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[44, 13]:
   ↪ Identifier 'dialogRef' is never reassigned; use 'const' instead of 'let
   ↪ '.
ERROR: /angular/src/app/error-message/error-message.component.spec.ts[25, 7]:
   ↪ Identifier 'fixture' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course/course.component.spec.ts[16, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-module/add-module.component.
   ↪ spec.ts[21, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-multiply-choice/add-multiply-
   ↪ choice.component.spec.ts[12, 9]: Identifier 'base' is never reassigned;
   ↪ use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/add-question/add-question.component
   ↪ .spec.ts[12, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.spec.ts[6, 24]: Too many spaces before 'from'
ERROR: /angular/src/app/learn/course-editor/create-course/create-course.
   ↪ component.spec.ts[19, 9]: Identifier 'base' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/dashboard/dashboard.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/module/module.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/multiple-choice-question/multiple-choice-question
   ↪ .component.spec.ts[15, 9]: Identifier 'base' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/question/question.component.spec.ts[14, 9]:
   ↪ Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/question/correct-feedback/correct-feedback.
   ↪ component.spec.ts[28, 9]: Identifier 'base' is never reassigned; use '
   ↪ const' instead of 'let'.
ERROR: /angular/src/app/learn/question/correct-feedback/correct-feedback.
   ↪ component.spec.ts[44, 13]: Identifier 'dialogRef' is never reassigned;
   ↪ use 'const' instead of 'let'.
ERROR: /angular/src/app/learn/question/correct-feedback/correct-feedback.
   ↪ component.spec.ts[25, 7]: Identifier 'fixture' is never reassigned; use
   ↪ 'const' instead of 'let'.
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
   ↪ spec.ts[28, 9]: Identifier 'base' is never reassigned; use 'const'
   ↪ instead of 'let'.
```

```
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
    ↪ spec.ts[44, 13]: Identifier 'dialogRef' is never reassigned; use 'const'
    ↪  instead of 'let'.
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
    ↪ spec.ts[25, 7]: Identifier 'fixture' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/loader/loader.component.spec.ts[12, 9]: Identifier '
    ↪ base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/login/login.component.spec.ts[13, 9]: Identifier 'base'
    ↪  is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/menu/menu.component.spec.ts[12, 9]: Identifier 'base'
    ↪ is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/page-not-found/page-not-found.component.spec.ts[12, 9]:
    ↪  Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/personal_statistics/statistics.component.spec.
    ↪ ts[13, 9]: Identifier 'base' is never reassigned; use 'const' instead of
    ↪  'let'.
ERROR: /angular/src/app/profile/profile-page/profile-page.component.spec.ts[12,
    ↪  9]: Identifier 'base' is never reassigned; use 'const' instead of 'let
    ↪ '.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.spec.ts[13,
    ↪ 9]: Identifier 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.
    ↪ spec.ts[12, 9]: Identifier 'base' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/register/register.component.spec.ts[12, 9]: Identifier
    ↪ 'base' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/service/user.service.spec.ts[6, 24]: " should be '

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: ‘ng lint‘
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_18_10_182Z-debug.log
```

pylint Output

```
************* Module learning_base.models
W: 59, 0: TODO: Refactor these to a decorator (fixme)
E: 36,16: Module 'django.utils.timezone' has no 'today' member (no-member)
E: 56,13: Module 'django.utils.timezone' has no 'localdate' member (no-member)
E:324,18: Module 'django.db.models' has no 'Course' member (no-member)
************* Module learning_base.views
W:301, 0: TODO: probably should be check_permissions(self, request) (fixme)
W:358, 0: TODO Implement saving a uorsers data (fixme)
W:419, 0: TODO: implement proper send_mail() (fixme)
W:451, 0: TODO: fix if an localization issues arrise (fixme)
C: 67,12: Invalid variable name "TYPES" (invalid-name)
C: 68,12: Invalid variable name "CATEGORIES" (invalid-name)
C: 69,12: Invalid variable name "LANGUAGES" (invalid-name)
C:165, 8: Invalid variable name "id" (invalid-name)
C:232, 8: Invalid variable name "e" (invalid-name)
E:452,34: Module 'django.utils.timezone' has no 'localdate' member (no-member)
C: 15, 0: Imports from package django are not grouped (ungrouped-imports)
************* Module learning_base.serializers
W:253, 0: TODO add language to profile (fixme)
W: 3, 0: Wildcard import models (wildcard-import)
W: 5, 0: Wildcard import learning_base.drag_and_drop.models (wildcard-import)
W: 7, 0: Wildcard import learning_base.drag_and_drop.serializer (wildcard-
    ↪ import)
C:186, 8: Invalid variable name "e" (invalid-name)
E:188,36: Instance of 'Exception' has no 'detail' member (no-member)
C:228, 8: Invalid variable name "p" (invalid-name)
W:248,-1: String statement has no effect (pointless-string-statement)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import apps from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 5, 0: Unused import DragAndDropQuestion from wildcard import (unused-
    ↪ wildcard-import)
W: 6, 0: Unused import b64decode from wildcard import (unused-wildcard-import)
W: 7, 0: Unused import DragAndDropSerializer from wildcard import (unused-
    ↪ wildcard-import)
C: 5, 0: external import "from learning_base.drag_and_drop.models import *"
    ↪ should be placed before "from .models import *" (wrong-import-order)
C: 7, 0: external import "from learning_base.drag_and_drop.serializer import *"
    ↪  should be placed before "from .models import *" (wrong-import-order)
************* Module learning_base.admin
W: 3, 0: Wildcard import learning_base.models (wildcard-import)
```

```
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import apps from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
************* Module learning_base.drag_and_drop.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 2, 0: Unused import DragAndDropQuestion from wildcard import (unused-
    ↪ wildcard-import)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)
************* Module learning_base.drag_and_drop.models
W: 1, 0: Unused models imported from django.db (unused-import)

------------------------------------------------------------------
Your code has been rated at 9.69/10 (previous run: 8.42/10, +1.27)
```

coverage Output

Coverage report: 74%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	11	0	0	100%
django/learning_base/drag_and_drop/__init__.py	0	0	0	100%
django/learning_base/drag_and_drop/models.py	4	0	0	100%
django/learning_base/drag_and_drop/serializer.py	4	0	0	100%
django/learning_base/models.py	106	20	0	81%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	38	11	0	71%
django/learning_base/multiple_choice/serializer.py	53	21	0	60%
django/learning_base/serializers.py	183	52	0	72%
django/learning_base/tests.py	159	0	0	100%
django/learning_base/views.py	230	103	0	55%
django/manage.py	13	6	0	54%
Total	**833**	**213**	**0**	**74%**

coverage.py v4.4.1, created at 2017-09-09 15:50

bandit Output

```
Run started:2017-09-12 10:28:17.042928

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 1987
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.7 Iteration 13 - 10.08.

Abgegebene Userstories: 24, 26, 28

Commit Hash: b5e9a319bf3bd0bc33e544ff615e1bd8b62c7931

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 24

Reviewer: Leonhard Wiedmann

Entwickler: Claas Völker

Commithash: 7af0e828 diff b5e9a319

Datum: 11.08.2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss f r die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?

  ja

- ☐ Sind Schaltfl chen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?

  ja

- ☐ Sind Buttons durch ihre Position direkt ersichtlich?

  ja

- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?

  ja

- ☐ Sind die Schaltfl chen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?

  ja

- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

  ja

1

- ☐ Ist alles in der Mindestaufl sung (1024x768) erkennbar?

  Ja

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☒ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☒ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☒ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☒ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☒ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  Nein

- ☐ Gibt es obsoleten Code?

  Nein

- ☐ Gibt es unbenutzte Variablen?

  Nein

- ☐ Wurden bereits vorhandene Funktionalit ten neu implementiert?

  Nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  ~~Keine~~ ja

- ☐ F r das Frontend:

  ja, keine neuen Post methoden

  - ☐ Sind alle POST Methoden richtig formatiert?

- ☐ F r das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    ja

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?

      ja

  - ☐ Geben alle Views einen g ltigen und passenden Status Code zur ck?

    ja

  - ☐ Wurde die Datenbankenstruktur ge ndert?

    ja

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?

      ja, neuer Fragetyp-

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ja

  - ☐ Autor

    ja

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ja

  - ☐ Autor

    ja

  - ☐ Eingabeparameter

    /

  - ☐ R ckgabewert

    /

- ☐ Gibt es weitere, schwer verst ndliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  Nein

- ☐ Sind die Kommentare verst ndlich f r alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollst ndig?

  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  ja

- ☐ Wie hoch ist die Statement Coverage?

  75 %

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 26

Reviewer: Claas
Entwickler: Leon
Commithash: 5ee0422 diff fdb3e8 (committed with 28)
Datum: 10.08.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  die externen Seiten von iGEL wurden nicht überprüft

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

  Backend wurde nicht geändert

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

  siehe oben

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  keine vorhanden

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  keine vorhanden

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  nein

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  nein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  na

- ☐ Für das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?

    gibt keine

- ☐ Für das Backend:

  keine Änderungen

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

  - ☐ Wurde die Datenbankenstruktur geändert?

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  ja

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung

  - ☑ Autor

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nope

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  code ist sehr einfach

- ☐ Ist die Dokumentation im Wiki vollständig?

  Es wurde nix geändert

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  Ja

- ☐ Wie hoch ist die Statement Coverage?

  75 % wird gefixed (kommt aus anderer US)

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 28

Reviewer: Claas
Entwickler: Leon
Commithash: 5ee0422 diff fdb3e8
Datum: 10.08.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- [x] Wurden nur Elemente des Material Designs verwendet?
- [x] Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- [x] Sind Buttons durch ihre Position direkt ersichtlich?
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- [x] Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- [x] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

ja

☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

ja

☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

nein -> wurde behoben

☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

ja (gab keine)

Checkliste Veränderbarkeit

Code Qualität

☐ Gibt es Meldungen der Tools?

ja (Linter errors) -> wurden direkt behoben

☐ Gibt es obsoleten Code?

nein

☐ Gibt es unbenutzte Variablen?

nein

☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  na

- ☐ Für das Frontend:

  - ☑ Sind alle POST Methoden richtig formatiert?

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    nein → Änderung war gut begründet

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      ja

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

    ja

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      /

Kommentare

- ☐ Sind alle Klassen kommentiert?

  ja

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor

- ☐ Sind alle Methoden kommentiert?

  Ja

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nein

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  Ja

- ☐ Ist die Dokumentation im Wiki vollständig?

  Ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  nein → wird gefixed

- ☐ Wie hoch ist die Statement Coverage?

  75% (wird gefixed)

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
npm info lifecycle clonecademy@0.0.0~lint: clonecademy@0.0.0

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/directive/logged-in.directive.ts[10, 31]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[2, 26]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[3, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[15, 34]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[16, 29]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[44, 11]:
    ↪ comment must start with a space
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[24, 1]:
    ↪ Exceeds maximum line length of 140
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[38, 5]:
    ↪ Forbidden 'var' keyword, use 'let' or 'const' instead
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[39, 5]:
    ↪ Forbidden 'var' keyword, use 'let' or 'const' instead
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[40, 5]:
    ↪ Forbidden 'var' keyword, use 'let' or 'const' instead
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[41, 5]:
    ↪ Forbidden 'var' keyword, use 'let' or 'const' instead
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[16, 41]:
    ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[27, 39]:
    ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[38, 9]:
    ↪ Identifier 'image' is never reassigned; use 'const' instead of 'var'.
```

```
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[39, 9]:
   ↪ Identifier 'file' is never reassigned; use 'const' instead of 'var'.
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[40, 9]:
   ↪ Identifier 'myReader' is never reassigned; use 'const' instead of 'var'.
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[41, 9]:
   ↪ Identifier 'that' is never reassigned; use 'const' instead of 'var'.
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[26, 39]: "
   ↪ should be '
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[27, 9]:
   ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[38, 15]:
   ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[39, 14]:
   ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[40, 18]:
   ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[42, 46]:
   ↪ missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[25, 3]: Declaration of instance field not allowed after
   ↪ declaration of instance method. Instead, this should come at the
   ↪ beginning of the class/interface.
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[19, 8]: Type string trivially inferred from a string
   ↪ literal, remove type annotation
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[25, 13]: Type string trivially inferred from a string
   ↪ literal, remove type annotation
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[21, 16]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[30, 18]: missing whitespace
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[3, 35]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[5, 25]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[19, 17]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[25, 22]: " should be '
ERROR: /angular/src/app/learn/course-editor/add-info-text/add-info-text.
   ↪ component.ts[33, 13]: " should be '
ERROR: /angular/src/app/learn/info-text/info-text.component.ts[16, 16]: missing
   ↪  whitespace
ERROR: /angular/src/app/learn/info-text/info-text.component.ts[3, 32]: " should
   ↪  be '
```

```
ERROR: /angular/src/app/learn/info-text/info-text.component.ts[4, 31]: " should
    ↪  be '
ERROR: /angular/src/app/learn/info-text/info-text.component.ts[7, 13]: The
    ↪ selector of the component "InformationTextComponent" should be named
    ↪ kebab-case and include dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/question-dictionary.ts[8, 49]: " should be '
ERROR: /angular/src/app/learn/question-dictionary.ts[14, 12]: " should be '
ERROR: /angular/src/app/learn/question-dictionary.ts[15, 12]: " should be '
ERROR: /angular/src/app/learn/question-dictionary.ts[18, 5]: " should be '
ERROR: /angular/src/app/learn/question-dictionary.ts[19, 5]: " should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[36, 5]: if
    ↪  statements must be braced
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[35, 17]:
    ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[38, 9]:
    ↪ Identifier 'request' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[4, 24]: "
    ↪ should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[23, 21]: "
    ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[39, 22]: "
    ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 38]:
    ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 39]:
    ↪ missing whitespace

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: `ng lint`
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_18_48_444Z-debug.log
```

pylint Output

```
************* Module learning_base.models
W: 48, 0: TODO: Implement correct user profile access string (fixme)
```

```
W: 59, 0: TODO: Refactor these to a decorator (fixme)
E: 36,16: Module 'django.utils.timezone' has no 'today' member (no-member)
E: 56,13: Module 'django.utils.timezone' has no 'localdate' member (no-member)
************* Module learning_base.views
W:301, 0: TODO: probably should be check_permissions(self, request) (fixme)
W:358, 0: TODO Implement saving a uorsers data (fixme)
W:475, 0: TODO: implement proper send_mail() (fixme)
W:507, 0: TODO: fix if an localization issues arrise (fixme)
W:546, 0: TODO Find out if it is usefull to send a 200 when a user was already
    ↪ mod (fixme)
C: 67,12: Invalid variable name "TYPES" (invalid-name)
C: 68,12: Invalid variable name "CATEGORIES" (invalid-name)
C: 69,12: Invalid variable name "LANGUAGES" (invalid-name)
C:165, 8: Invalid variable name "id" (invalid-name)
R:210, 8: Consider using ternary (question.module.is_first_module if question.
    ↪ is_first_question else request.user.try_set.filter(question__order=(
    ↪ question.order) - (1), solved=True).exists()) (consider-using-ternary)
C:245, 8: Invalid variable name "e" (invalid-name)
E:508,34: Module 'django.utils.timezone' has no 'localdate' member (no-member)
C: 15, 0: Imports from package django are not grouped (ungrouped-imports)
************* Module learning_base.serializers
W:252, 0: TODO add language to profile (fixme)
W: 3, 0: Wildcard import models (wildcard-import)
W: 5, 0: Wildcard import learning_base.info.serializer (wildcard-import)
C:185, 8: Invalid variable name "e" (invalid-name)
E:187,36: Instance of 'Exception' has no 'detail' member (no-member)
C:227, 8: Invalid variable name "p" (invalid-name)
W: 3, 0: Unused import apps from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 4, 0: Unused import b64decode from wildcard import (unused-wildcard-import)
W: 5, 0: Unused import InformationText from wildcard import (unused-wildcard-
    ↪ import)
C: 5, 0: external import "from learning_base.info.serializer import *" should
    ↪ be placed before "from .models import *" (wrong-import-order)
************* Module learning_base.admin
W: 3, 0: Wildcard import learning_base.models (wildcard-import)
W: 5, 0: Wildcard import learning_base.info.models (wildcard-import)
W: 3, 0: Unused import apps from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
```

```
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import User from wildcard import (unused-wildcard-import)
************* Module learning_base.drag_and_drop.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 2, 0: Unused import DragAndDropQuestion from wildcard import (unused-
    ↪ wildcard-import)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)
************* Module learning_base.drag_and_drop.models
W: 1, 0: Unused models imported from django.db (unused-import)
************* Module learning_base.info.serializer
W: 3, 0: Wildcard import models (wildcard-import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
************* Module learning_base.info.models
E: 34,15: Module 'learning_base.info.serializer' has no '
    ↪ InformationTextEditSerializer' member (no-member)

------------------------------------------------------------------
Your code has been rated at 9.65/10 (previous run: 6.83/10, +2.82)
```

coverage Output

Coverage report: 75%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	13	0	0	100%
django/learning_base/drag_and_drop/__init__.py	0	0	0	100%
django/learning_base/info/__init__.py	0	0	0	100%
django/learning_base/info/models.py	17	5	0	71%
django/learning_base/info/serializer.py	7	0	0	100%
django/learning_base/models.py	106	20	0	81%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	38	9	0	76%
django/learning_base/multiple_choice/serializer.py	53	7	0	87%
django/learning_base/serializers.py	181	50	0	72%
django/learning_base/tests.py	169	0	0	100%
django/learning_base/views.py	261	129	0	51%
django/manage.py	13	6	0	54%
Total	**890**	**226**	**0**	**75%**

coverage.py v4.4.1, created at 2017-09-09 15:50

Datensicherheit

bandit Output

```
Run started:2017-09-12 10:28:26.701400

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 1760
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.8 Iteration 14 - 17.08.

Abgegebene Userstories: 25,27

Commit Hash: ba0fab4c21d736b0f67f8fe21ef90c0da810ac4b

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US - 25

Reviewer: ilhan
Entwickler: Claas
Commithash: 5a0fa
Datum: 19.09

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools? Nein
- ☐ Gibt es obsoleten Code? Nein
- ☐ Gibt es unbenutzte Variablen? Nein
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?

    Fragestellung Info
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  alles Vollständig!

  Dies umfasst:
  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☑ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☑ Sind die Kommentare verständlich für alle Entwickler?
- ☑ Ist die Dokumentation im Wiki vollständig?

Tests

- ☑ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
npm info lifecycle clonecademy@0.0.0~lint: clonecademy@0.0.0

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/directive/logged-in.directive.ts[10, 31]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[2, 26]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[3, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[15, 34]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[16, 29]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 7]: missing
    ↪ whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[24, 1]:
    ↪ Exceeds maximum line length of 140
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[39, 5]:
    ↪ Forbidden 'var' keyword, use 'let' or 'const' instead
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[40, 5]:
    ↪ Forbidden 'var' keyword, use 'let' or 'const' instead
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[41, 5]:
    ↪ Forbidden 'var' keyword, use 'let' or 'const' instead
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[42, 5]:
    ↪ Forbidden 'var' keyword, use 'let' or 'const' instead
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[16, 41]:
    ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[28, 39]:
    ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[39, 9]:
    ↪ Identifier 'image' is never reassigned; use 'const' instead of 'var'.
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[40, 9]:
    ↪ Identifier 'file' is never reassigned; use 'const' instead of 'var'.
```

```
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[41, 9]:
   ↪ Identifier 'myReader' is never reassigned; use 'const' instead of 'var'.
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[42, 9]:
   ↪ Identifier 'that' is never reassigned; use 'const' instead of 'var'.
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[27, 39]: "
   ↪ should be '
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[28, 9]:
   ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[39, 15]:
   ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[40, 14]:
   ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[41, 18]:
   ↪ missing whitespace
ERROR: /angular/src/app/image-cropper/image-cropper.component.ts[43, 46]:
   ↪ missing whitespace
ERROR: /angular/src/app/learn/question-dictionary.ts[8, 49]: " should be '
ERROR: /angular/src/app/learn/question-dictionary.ts[14, 12]: " should be '
ERROR: /angular/src/app/learn/question-dictionary.ts[15, 12]: " should be '
ERROR: /angular/src/app/learn/question-dictionary.ts[18, 5]: " should be '
ERROR: /angular/src/app/learn/question-dictionary.ts[19, 5]: " should be '
ERROR: /angular/src/app/directive/module.directive.ts[10, 29]: missing
   ↪ whitespace
ERROR: /angular/src/app/learn/question/wrong-feedback/wrong-feedback.component.
   ↪ ts[14, 9]: Type string trivially inferred from a string literal, remove
   ↪ type annotation
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[36, 5]: if
   ↪  statements must be braced
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[35, 17]:
   ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[38, 9]:
   ↪ Identifier 'request' is never reassigned; use 'const' instead of 'let'.
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[4, 24]: "
   ↪ should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[23, 21]: "
   ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[39, 22]: "
   ↪  should be '
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 38]:
   ↪ missing whitespace
ERROR: /angular/src/app/profile/request-mod/request-mod.component.ts[40, 39]:
   ↪ missing whitespace
ERROR: /angular/src/app/static-page/static-page.component.ts[21, 20]: missing
   ↪ whitespace
ERROR: /angular/src/app/static-page/static-page.component.ts[17, 28]: " should
   ↪ be '
```

```
ERROR: /angular/src/app/learn/view-courses/view-courses.component.ts[15, 33]:
    ↪ missing whitespace
ERROR: /angular/src/app/learn/view-courses/view-courses.component.ts[19, 20]:
    ↪ missing whitespace
ERROR: /angular/src/app/learn/view-courses/view-courses.component.ts[19, 3]:
    ↪ Implement lifecycle hook interface AfterViewInit for method
    ↪ ngAfterViewInit in class CourseViewComponent (https://goo.gl/w1Nwk3)

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: ‘ng lint‘
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_19_08_321Z-debug.log
```

pylint Output

```
************* Module learning_base.models
W: 47, 0: TODO: Implement correct user profile access string (fixme)
W: 58, 0: TODO: Refactor these to a decorator (fixme)
E: 35,16: Module 'django.utils.timezone' has no 'today' member (no-member)
E: 55,17: Module 'django.utils.timezone' has no 'localdate' member (no-member)
************* Module learning_base.views
W:337, 0: TODO: probably should be check_permissions(self, request) (fixme)
W:520, 0: TODO: implement proper send_mail() (fixme)
W:552, 0: TODO: fix if an localization issues arrise (fixme)
C: 65,12: Invalid variable name "TYPES" (invalid-name)
C: 66,12: Invalid variable name "CATEGORIES" (invalid-name)
C: 67,12: Invalid variable name "LANGUAGES" (invalid-name)
C: 93, 8: Invalid variable name "e" (invalid-name)
C:170, 8: Invalid variable name "id" (invalid-name)
C:502, 8: Invalid variable name "serializeData" (invalid-name)
E:553,34: Module 'django.utils.timezone' has no 'localdate' member (no-member)
C: 13, 0: Imports from package django are not grouped (ungrouped-imports)
************* Module learning_base.serializers
W:302, 0: TODO add language to profile (fixme)
W: 5, 0: Wildcard import models (wildcard-import)
W: 7, 0: Wildcard import learning_base.info.serializer (wildcard-import)
```

```
C:132,12: Invalid variable name "queryQuestions" (invalid-name)
C:134,16: Invalid variable name "q" (invalid-name)
C:214,12: Invalid variable name "moduleQuery" (invalid-name)
C:229, 8: Invalid variable name "e" (invalid-name)
C:277, 8: Invalid variable name "p" (invalid-name)
W: 5, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 5, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 5, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 5, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 6, 0: Unused import b64decode from wildcard import (unused-wildcard-import)
W: 7, 0: Unused import InformationText from wildcard import (unused-wildcard-
    ↪ import)
C: 7, 0: external import "from learning_base.info.serializer import *" should
    ↪ be placed before "from .models import *" (wrong-import-order)
************* Module learning_base.admin
W: 3, 0: Wildcard import learning_base.models (wildcard-import)
W: 5, 0: Wildcard import learning_base.info.models (wildcard-import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
************* Module learning_base.drag_and_drop.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import DragAndDropQuestion from wildcard import (unused-
    ↪ wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)
************* Module learning_base.drag_and_drop.models
W: 1, 0: Unused models imported from django.db (unused-import)
************* Module learning_base.info.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)

------------------------------------------------------------------
Your code has been rated at 9.61/10 (previous run: 7.83/10, +1.74)
```

coverage Output

Coverage report: 81%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	13	0	0	100%
django/learning_base/drag_and_drop/__init__.py	0	0	0	100%
django/learning_base/info/__init__.py	0	0	0	100%
django/learning_base/info/models.py	20	6	0	70%
django/learning_base/info/serializer.py	11	0	0	100%
django/learning_base/models.py	104	14	0	87%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	38	5	0	87%
django/learning_base/multiple_choice/serializer.py	53	1	0	98%
django/learning_base/serializers.py	209	39	0	81%
django/learning_base/tests.py	249	4	0	98%
django/learning_base/views.py	273	113	0	59%
django/manage.py	13	6	0	54%
Total	**1015**	**188**	**0**	**81%**

coverage.py v4.4.1, created at 2017-09-09 15:53

Datensicherheit

bandit Output

```
Run started:2017-09-12 10:28:36.937174

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 2189
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.9 Iteration 15 - 24.08.

Abgegebene Userstories: 38

Commit Hash: 78f8954ebc159fbcb6ed6a767e18d3437a604d22

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US -38

Reviewer: Ilkan
Entwickler: Claas
Commithash: 78f45
Datum: 25.04

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools? Nein
- ☐ Gibt es obsoleten Code? Nein
- ☐ Gibt es unbenutzte Variablen? Nein
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

- ☑ Wurden Hilfsmethoden ausgelagert? Ja
- ☑ Für das Frontend:
  - ☑ Sind alle POST Methoden richtig formatiert? Keine
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Warum?
Vollständig ?

Kommentare

- ☑ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☑ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☑ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

Tests

- ☑ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
npm info lifecycle clonecademy@0.0.0~lint: clonecademy@0.0.0

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 41]: missing
    ↪ whitespace
ERROR: /angular/src/app/directive/logged-in.directive.ts[2, 26]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[3, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 24]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[15, 34]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[16, 29]: " should be '
ERROR: /angular/src/app/directive/logged-in.directive.ts[14, 7]: missing
    ↪ whitespace

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: `ng lint`
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_20_41_339Z-debug.log
```

pylint Output

```
************* Module learning_base.models
W: 47, 0: TODO: Implement correct user profile access string (fixme)
```

```
W: 58, 0: TODO: Refactor these to a decorator (fixme)
E: 35,16: Module 'django.utils.timezone' has no 'today' member (no-member)
E: 55,17: Module 'django.utils.timezone' has no 'localdate' member (no-member)
************* Module learning_base.views
W:125, 0: TODO: Try if the conformate "ans" instead of "error" works as good as
    ↪  this (fixme)
W:338, 0: TODO: probably should be check_permissions(self, request) (fixme)
W:521, 0: TODO: implement proper send_mail() (fixme)
W:553, 0: TODO: fix if an localization issues arrise (fixme)
W:596, 0: TODO Find out if it is useful to send a 200 when a user (fixme)
C: 65,12: Invalid variable name "TYPES" (invalid-name)
C: 66,12: Invalid variable name "CATEGORIES" (invalid-name)
C: 67,12: Invalid variable name "LANGUAGES" (invalid-name)
C: 93, 8: Invalid variable name "e" (invalid-name)
C:171, 8: Invalid variable name "id" (invalid-name)
C:503, 8: Invalid variable name "serializeData" (invalid-name)
E:554,34: Module 'django.utils.timezone' has no 'localdate' member (no-member)
C: 12, 0: Imports from package rest_framework are not grouped (ungrouped-
    ↪ imports)
C: 13, 0: Imports from package django are not grouped (ungrouped-imports)
************* Module learning_base.serializers
W:302, 0: TODO add language to profile (fixme)
W: 5, 0: Wildcard import models (wildcard-import)
W: 7, 0: Wildcard import learning_base.info.serializer (wildcard-import)
C:132,12: Invalid variable name "queryQuestions" (invalid-name)
C:134,16: Invalid variable name "q" (invalid-name)
C:214,12: Invalid variable name "moduleQuery" (invalid-name)
C:229, 8: Invalid variable name "e" (invalid-name)
C:277, 8: Invalid variable name "p" (invalid-name)
W: 3, 0: Unused status imported from rest_framework (unused-import)
W: 5, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 5, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 5, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 5, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 6, 0: Unused import b64decode from wildcard import (unused-wildcard-import)
W: 7, 0: Unused import InformationText from wildcard import (unused-wildcard-
    ↪ import)
C: 7, 0: external import "from learning_base.info.serializer import *" should
    ↪ be placed before "from .models import *" (wrong-import-order)
************* Module learning_base.admin
W: 3, 0: Wildcard import learning_base.models (wildcard-import)
W: 5, 0: Wildcard import learning_base.info.models (wildcard-import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
```

```
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
************* Module learning_base.drag_and_drop.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 2, 0: Unused import DragAndDropQuestion from wildcard import (unused-
    ↪ wildcard-import)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)
************* Module learning_base.drag_and_drop.models
W: 1, 0: Unused models imported from django.db (unused-import)
************* Module learning_base.info.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)

------------------------------------------------------------------
Your code has been rated at 9.62/10 (previous run: 8.23/10, +1.39)
```

coverage Output

Coverage report: 81%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	13	0	0	100%
django/learning_base/drag_and_drop/__init__.py	0	0	0	100%
django/learning_base/info/__init__.py	0	0	0	100%
django/learning_base/info/models.py	20	6	0	70%
django/learning_base/info/serializer.py	11	0	0	100%
django/learning_base/models.py	104	14	0	87%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	38	5	0	87%
django/learning_base/multiple_choice/serializer.py	53	1	0	98%
django/learning_base/serializers.py	209	39	0	81%
django/learning_base/tests.py	249	4	0	98%
django/learning_base/views.py	273	113	0	59%
django/manage.py	13	6	0	54%
Total	**1015**	**188**	**0**	**81%**

coverage.py v4.4.1, created at 2017-09-09 15:54

bandit Output

```
Run started:2017-09-12 10:28:47.526662

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 2189
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.10 Iteration 16 - 31.08.

Abgegebene Userstories: 30, 32, 36, 37, 40, 41

Commit Hash: 11bc7fa37a20ac045d5a6c29e4e3a060b73b3504

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US - 30

Reviewer: ilkan
Entwickler: Tobi
Commithash: 11bc7
Datum: 01.05

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

02.05

- [x] Wurden nur Elemente des Material Designs verwendet? teils, bessern...
- [x] Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- [x] Sind Buttons durch ihre Position direkt ersichtlich?
  ↳ Buttons springen bei auswahl
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
  ↳ zu viel unnötige Info besser als Hint am button
- [x] Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- [x] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☑ Gibt es Meldungen der Tools? Nein
- ☑ Gibt es obsoleten Code? Nein
- ☑ Gibt es unbenutzte Variablen? Nein
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

- ☐ Wurden Hilfsmethoden ausgelagert?

- ☐ Für das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

  - ☐ Wurde die Datenbankenstruktur geändert?

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

01.05

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

  - ☐ Autor

02.05

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☑ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☑ Sind die Kommentare verständlich für alle Entwickler?

  Keine Kommentare!

- ☐ Ist die Dokumentation im Wiki vollständig?

Tests

- ☑ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 32

Reviewer: Leonhard Wiedmann
Entwickler: Tobias Huber
Commithash: 11be7 fa37 diff ffc8efbf70
Datum: 07.09

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss f r die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?
  ja
- ☐ Sind Schaltfl chen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
  siehe Checkliste US 31
- ☐ Sind Buttons durch ihre Position direkt ersichtlich?
  ja
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
  siehe Checkliste US 31
- ☐ Sind die Schaltfl chen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
  ja
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?
  ja

1

☒ Ist alles in der Mindestaufl sung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

☒ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

☒ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

☒ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

☒ Werden alle Nutzereingaben nach Fehlern gefiltert?

☒ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

☐ Gibt es Meldungen der Tools?

Nein

☐ Gibt es obsoleten Code?

Nein

☐ Gibt es unbenutzte Variablen?

Nein

☐ Wurden bereits vorhandene Funktionalit ten neu implementiert?

ja, wird nachgebessert

2

- ☐ Wurden Hilfsmethoden ausgelagert?
  ja

- ☐ F r das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?
    ja

- ☐ F r das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    ja

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?
      ja

  - ☐ Geben alle Views einen g ltigen und passenden Status Code zur ck?
    ja

  - ☐ Wurde die Datenbankenstruktur ge ndert?
    Nein

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
    ja

  - ☐ Autor
    ja

- ☐ Sind alle Methoden kommentiert?

  ja

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ja

  - ☐ Autor

    ja

  - ☐ Eingabeparameter

    ja

  - ☐ R ckgabewert

    /

- ☐ Gibt es weitere, schwer verst ndliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  Nein

- ☐ Sind die Kommentare verst ndlich f r alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollst ndig?

  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  ja

- ☐ Wie hoch ist die Statement Coverage?

  80 %

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 37

Reviewer: Claas
Entwickler: Leo
Commithash: d6dac19 diff 563946
Datum: 31.08

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- [x] Wurden nur Elemente des Material Designs verwendet?
- [x] Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- [x] Sind Buttons durch ihre Position direkt ersichtlich?
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- [x] Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- [x] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

keinerlei Inputs / keine Änderungen im Backend

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  nein

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  nein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?

    gab keine
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☑ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

nur HTML/CSS

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

keine Schnittstelle

Tests

- ☐ Sind Tests aller Methoden vorhanden?

keine zu testenden Methoden

- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 40

Reviewer: Leonhard Wiedmann

Entwickler: Claas Volcker

Commithash: 78f8955e diff 11bc7fa3

Datum: 31.08

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss f r die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?

  ja

- ☐ Sind Schaltfl chen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?

  keine neuen Schaltfläche

- ☐ Sind Buttons durch ihre Position direkt ersichtlich?

  keine neuen Buttons

- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?

  ja

- ☐ Sind die Schaltfl chen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?

  /

- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

  ja

1

- ☐ Ist alles in der Mindestaufl sung (1024x768) erkennbar?

  ja

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☒ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☒ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☒ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  keine
- ☒ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  Nein
- ☐ Gibt es obsoleten Code?

  Nein
- ☐ Gibt es unbenutzte Variablen?

  Nein
- ☐ Wurden bereits vorhandene Funktionalit ten neu implementiert?

  Nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  Keine Hilfsmethoden benötigt

- ☐ F r das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?

    Keine Post methoden

- ☐ F r das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    ja

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?

      ja

  - ☐ Geben alle Views einen g ltigen und passenden Status Code zur ck?

    ja

  - ☐ Wurde die Datenbankenstruktur ge ndert?

    nein

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ja

  - ☐ Autor

    ja

- ☐ Sind alle Methoden kommentiert?

  ja

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ja

  - ☐ Autor

    ja

  - ☐ Eingabeparameter

    ja

  - ☐ R ckgabewert

    ja

- ☐ Gibt es weitere, schwer verst ndliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  Nein

- ☐ Sind die Kommentare verst ndlich f r alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollst ndig?

  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  ja

- ☐ Wie hoch ist die Statement Coverage?

  80 %

Veränderbarkeit

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
[..................] / : info lifecycle clonecademy@0.0.0~prelint: clonecademy@

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [41, 7]: comment must start with a space
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [61, 11]: comment must start with a space
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [16, 14]: Type boolean trivially inferred from a boolean literal, remove
    ↪  type annotation
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [34, 18]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [35, 22]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [40, 13]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [64, 36]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [71, 9]: misplaced 'else'
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [67, 42]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [74, 28]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [75, 55]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [76, 70]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [77, 54]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [66, 15]: Identifier 'lastQuestion' is never reassigned; use 'const'
    ↪ instead of 'let'.
```

```
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [63, 13]: Identifier 'lastModule' is never reassigned; use 'const'
    ↪ instead of 'let'.
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [53, 19]: " should be '
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [55, 40]: " should be '
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [87, 29]: " should be '
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [64, 23]: != should be !==
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [67, 34]: == should be ===
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [55, 33]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [78, 37]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [78, 38]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [78, 42]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [78, 43]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [23, 23]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [23, 28]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [35, 7]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [64, 11]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [67, 13]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [74, 11]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [75, 14]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [76, 16]: missing whitespace
ERROR: /angular/src/app/learn/question-sidenav/question-sidenav.component.ts
    ↪ [77, 17]: missing whitespace

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
```

```
npm ERR! clonecademy@0.0.0 lint: ‘ng lint‘
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_21_02_936Z-debug.log
```

pylint Output

```
************* Module learning_base.models
W: 58, 0: TODO: Refactor these to a decorator (fixme)
E: 35,16: Module ’django.utils.timezone’ has no ’today’ member (no-member)
E: 55,16: Module ’django.utils.timezone’ has no ’localdate’ member (no-member)
************* Module learning_base.views
W:124, 0: TODO: Try if the conformate "ans" instead of "error" (fixme)
W:521, 0: TODO: implement proper send_mail() (fixme)
W:553, 0: TODO: fix if an localization issues arrise (fixme)
C: 64,12: Invalid variable name "TYPES" (invalid-name)
C: 65,12: Invalid variable name "CATEGORIES" (invalid-name)
C: 66,12: Invalid variable name "LANGUAGES" (invalid-name)
C: 92, 8: Invalid variable name "e" (invalid-name)
C:171, 8: Invalid variable name "id" (invalid-name)
C:503, 8: Invalid variable name "serializeData" (invalid-name)
E:554,34: Module ’django.utils.timezone’ has no ’localdate’ member (no-member)
W: 11, 0: Unused Profile imported from models (unused-import)
C: 10, 0: external import "from learning_base import custom_permissions" should
    ↪  be placed before "from . import serializers as serializer" (wrong-
    ↪ import-order)
C: 13, 0: external import "from django.core.mail import send_mail" should be
    ↪ placed before "from . import serializers as serializer" (wrong-import-
    ↪ order)
C: 15, 0: external import "from django.utils import timezone" should be placed
    ↪ before "from . import serializers as serializer" (wrong-import-order)
C: 13, 0: Imports from package django are not grouped (ungrouped-imports)
************* Module learning_base.serializers
W:313, 0: TODO add language to profile (fixme)
W: 8, 0: Wildcard import models (wildcard-import)
C: 48,12: Invalid variable name "m" (invalid-name)
C:156,16: Invalid variable name "q" (invalid-name)
C:257, 8: Invalid variable name "e" (invalid-name)
C:306, 8: Invalid variable name "p" (invalid-name)
```

```
W: 8, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 8, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 8, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 8, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
************* Module learning_base.admin
W: 3, 0: Wildcard import models (wildcard-import)
W: 5, 0: Wildcard import info.models (wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
************* Module learning_base.info.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 30, 4: String statement has no effect (pointless-string-statement)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)
C: 4, 0: standard import "from re import compile" should be placed before "from
    ↪  rest_framework import serializers" (wrong-import-order)

------------------------------------------------------------------
Your code has been rated at 9.66/10 (previous run: 9.47/10, +0.19)
```

coverage Output

Coverage report: 80%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	14	0	0	100%
django/learning_base/custom_permissions.py	11	1	0	91%
django/learning_base/info/__init__.py	0	0	0	100%
django/learning_base/info/models.py	38	14	0	63%
django/learning_base/info/serializer.py	23	5	0	78%
django/learning_base/models.py	114	20	0	82%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	38	5	0	87%
django/learning_base/multiple_choice/serializer.py	53	1	0	98%
django/learning_base/serializers.py	222	43	0	81%
django/learning_base/tests.py	294	4	0	99%
django/learning_base/views.py	292	129	0	56%
django/manage.py	13	6	0	54%
Total	**1144**	**228**	**0**	**80%**

coverage.py v4.4.1, created at 2017-09-09 15:55

Datensicherheit

bandit Output

```
Run started:2017-09-12 10:28:58.552596

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 2422
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.11 Iteration 17 - 07.09.

Abgegebene Userstories: 31, 33, 34, 42, 43, 44, 46

Commit Hash: ffc8afbf7012f4b2d63e5b58ac8f0c224a7ee382

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 31

Reviewer: Leonhard Wiedmann
Entwickler: Tobias Huber
Commithash: 11bc7fa37 diff ffc8afbf70
Datum: 07.09

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss f r die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?
  ja
- ☐ Sind Schaltfl chen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
  Die Anordnung der Buttons ist nicht zusammen in div Containern und ~~nach aktiv~~ wird nach Aktivieren verschoben. wird noch gebessert.
- ☐ Sind Buttons durch ihre Position direkt ersichtlich?
  ja
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
  Es gibt einen sehr langen Beschreibungstext, der es unübersichtlich macht. Wird durch einen Tooltip ersetzt.
- ☐ Sind die Schaltfl chen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
  ja
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?
  ja

1

- ☐ Ist alles in der Mindestaufl sung (1024x768) erkennbar?

ja

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☒ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☒ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☒ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☒ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☒ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

Nein

- ☐ Gibt es obsoleten Code?

Nein

- ☐ Gibt es unbenutzte Variablen?

Nein

- ☐ Wurden bereits vorhandene Funktionalit ten neu implementiert?

ja, wird nachgebessert

- ☐ Wurden Hilfsmethoden ausgelagert?
  ja

- ☐ F r das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?
    ja

- ☐ F r das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    ja

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?
      ja

  - ☐ Geben alle Views einen g ltigen und passenden Status Code zur ck?
    ja

  - ☐ Wurde die Datenbankenstruktur ge ndert?
    Nein

    - ☐ Wenn ja: Sind diese begr ndet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?
  ja

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
    ja

  - ☐ Autor
    ja

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

    ja

  - ☐ Autor

    ja

  - ☐ Eingabeparameter

    Nein, wird nachgebessert

  - ☐ R ckgabewert

    keine Vorhanden

- ☐ Gibt es weitere, schwer verst ndliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  Nein

- ☐ Sind die Kommentare verst ndlich f r alle Entwickler?

  ja

- ☐ Ist die Dokumentation im Wiki vollst ndig?

  ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  ja

- ☐ Wie hoch ist die Statement Coverage?

  80 %

4

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 33 (Kategorie bearbeiten)

Reviewer: Claas
Entwickler: Tobias
Commithash:
Datum: 07.09.2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss f r die Elemente folgende Checkliste beachtet werden:

- [x] Wurden nur Elemente des Material Designs verwendet?
- [x] Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- [x] Sind Buttons durch ihre Position direkt ersichtlich?
- [x] Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- [x] Sind die Schaltfl chen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- [x] Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  Linter errors wurde behoben

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  nein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  nein

- ☑ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☑ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☑ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☑ Wenn ja: Sind diese begründet und dokumentiert?
  - ☑ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?

    nein
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☑ Sind alle Klassen kommentiert?

  Dies umfasst:
  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nein

- ☑ Sind die Kommentare verständlich für alle Entwickler?

- ☑ Ist die Dokumentation im Wiki vollst ndig?

Tests

- ☑ Sind Tests aller Methoden vorhanden?

- ☐ Wie hoch ist die Statement Coverage?

  80%

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US -34

Reviewer: ilkan
Entwickler: Tobi
Commithash: ffc8a
Datum: 09.05

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?

  nach Besserung wie in der Gruppe abgesprochen.
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools? Nein
- ☐ Gibt es obsoleten Code? Nein
- ☐ Gibt es unbenutzte Variablen? Nein
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentar alle vollständig!

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☑ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☑ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

Tests

- ☑ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US – 42

Reviewer: i'lhas
Entwickler: Clucas
Commithash: ffc5a
Datum: 08.05

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☑ Gibt es Meldungen der Tools? Nein
- ☑ Gibt es obsoleten Code? Nein
- ☑ Gibt es unbenutzte Variablen? Nein
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

- ☑ Wurden Hilfsmethoden ausgelagert? keine HM
- ☑ Für das Frontend:
  - ☑ Sind alle POST Methoden richtig formatiert? keine Postmethoden
- ☑ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

alle Kommentare vollständig ☑

Kommentare

- ☑ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☑ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☑ Sind die Kommentare verständlich für alle Entwickler?
- ☑ Ist die Dokumentation im Wiki vollständig? Keine

Tests

- ☑ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US - 46

Reviewer: ilhan
Entwickler: Claas
Commithash: ffc8a
Datum: 08.05

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art? Ranking Tabell etwas verrutscht dargestellt.
- ☑ Sind Buttons durch ihre Position direkt ersichtlich? keine Buttons
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☑ Gibt es Meldungen der Tools? Nein
- ☑ Gibt es obsoleten Code? Nein
- ☑ Gibt es unbenutzte Variablen? Nein
- ☑ Wurden bereits vorhandene Funktionalitäten neu implementiert? Nein

- ☐ Wurden Hilfsmethoden ausgelagert? Ja
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    ↳ ~~[illegible]~~ änderungen: Rankings hinzugefügt
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare Vollständig

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

Tests

- ☐ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Veränderbarkeit

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
[..................] / : info lifecycle clonecademy@0.0.0~prelint: clonecademy@

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/learn/info-text/info-text.component.ts[7, 13]: The
    ↪ selector of the component "InformationTextComponent" should be named
    ↪ kebab-case and include dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/info-youtube/info-youtube.component.ts[10, 13]:
    ↪ The selector of the component "InformationYoutubeComponent" should be
    ↪ named kebab-case and include dash (https://goo.gl/mBg67Z)
ERROR: /angular/src/app/learn/info-youtube/info-youtube.component.ts[18, 3]:
    ↪ Implement lifecycle hook interface OnInit for method ngOnInit in class
    ↪ InformationYoutubeComponent (https://goo.gl/w1Nwk3)
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [30, 16]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 78]: missing whitespace
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [32, 13]: Identifier 'data' is never reassigned; use 'const' instead of
    ↪ 'let'.
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [6, 29]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 56]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 68]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 85]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [18, 97]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 37]: " should be '
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [31, 9]: missing whitespace
```

```
ERROR: /angular/src/app/profile/user-detail-user/user-detail-user.component.ts
    ↪ [27, 5]: Implement lifecycle hook interface OnInit for method ngOnInit
    ↪ in class UserDetailUserComponent (https://goo.gl/w1Nwk3)

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: ‘ng lint‘
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_21_31_490Z-debug.log
```

pylint Output

```
************* Module learning_base.models
W: 63, 0: TODO: Refactor these to a decorator (fixme)
E: 44,16: Module ’django.utils.timezone’ has no ’today’ member (no-member)
E: 60,12: Module ’django.utils.timezone’ has no ’localdate’ member (no-member)
************* Module learning_base.views
W:125, 0: TODO: Try if the conformate "ans" instead of "error" (fixme)
W:633, 0: TODO: implement proper send_mail() (fixme)
W:666, 0: TODO: fix if an localization issues arrise (fixme)
C: 64,12: Invalid variable name "TYPES" (invalid-name)
C: 65,12: Invalid variable name "CATEGORIES" (invalid-name)
C: 66,12: Invalid variable name "LANGUAGES" (invalid-name)
C:172, 8: Invalid variable name "id" (invalid-name)
C:292, 8: Invalid variable name "e" (invalid-name)
C:320,16: Invalid variable name "nextType" (invalid-name)
C:588, 8: Invalid variable name "serializeData" (invalid-name)
E:667,34: Module ’django.utils.timezone’ has no ’localdate’ member (no-member)
C: 10, 0: external import "from learning_base import custom_permissions" should
    ↪  be placed before "from . import serializers as serializer" (wrong-
    ↪ import-order)
C: 13, 0: external import "from django.core.mail import send_mail" should be
    ↪ placed before "from . import serializers as serializer" (wrong-import-
    ↪ order)
C: 15, 0: external import "from django.utils import timezone" should be placed
    ↪ before "from . import serializers as serializer" (wrong-import-order)
C: 13, 0: Imports from package django are not grouped (ungrouped-imports)
```

```
************* Module learning_base.serializers
W:422, 0: TODO add language to profile (fixme)
C: 53,12: Invalid variable name "m" (invalid-name)
C:252,24: Invalid variable name "q" (invalid-name)
C:262,12: Invalid variable name "e" (invalid-name)
C:414, 8: Invalid variable name "p" (invalid-name)
************* Module learning_base.admin
W: 3, 0: Wildcard import models (wildcard-import)
W: 5, 0: Wildcard import info.models (wildcard-import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import CourseManager from wildcard import (unused-wildcard-
    ↪ import)
************* Module learning_base.info.serializer
W: 2, 0: Wildcard import models (wildcard-import)
W: 30, 4: String statement has no effect (pointless-string-statement)
W: 2, 0: Unused import Question from wildcard import (unused-wildcard-import)
W: 2, 0: Unused import models from wildcard import (unused-wildcard-import)
C: 4, 0: standard import "from re import compile" should be placed before "from
    ↪  rest_framework import serializers" (wrong-import-order)

------------------------------------------------------------------
Your code has been rated at 9.76/10 (previous run: 5.20/10, +4.56)
```

coverage Output

Coverage report: 80%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	16	0	0	100%
django/learning_base/custom_permissions.py	14	2	0	86%
django/learning_base/info/__init__.py	0	0	0	100%
django/learning_base/info/models.py	42	16	0	62%
django/learning_base/info/serializer.py	23	5	0	78%
django/learning_base/models.py	150	31	0	79%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	40	5	0	88%
django/learning_base/multiple_choice/serializer.py	53	1	0	98%
django/learning_base/serializers.py	303	61	0	80%
django/learning_base/tests.py	351	5	0	99%
django/learning_base/views.py	354	152	0	57%
django/manage.py	13	6	0	54%
Total	**1391**	**284**	**0**	**80%**

coverage.py v4.4.1, created at 2017-09-09 15:57

bandit Output

```
Run started:2017-09-12 10:29:10.138068

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 3312
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.12 Iteration 18 - 14.09.

Abgegebene Userstories: 39, 45, 47, 49, 50, 51, 54, 55

Commit Hash:

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 39

Reviewer: Tobias

Entwickler: Ichan

Commithash: de87a0d

Datum: 14.09.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet? ✓
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art? ✓
- ☐ Sind Buttons durch ihre Position direkt ersichtlich? ✓
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?

Progressbalken wird verdeckt! (geändert) ✓

- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben? ✓

~~Klasse~~

- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen? ✓

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar? ✓

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt: keine Backend Änderungen

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

klassische Fehler ("statt') wurden behoben

- ☐ Gibt es obsoleten Code? X
- ☐ Gibt es unbenutzte Variablen? X
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? X

- ☐ Wurden Hilfsmethoden ausgelagert?

  X

- ☐ Für das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert? ✓

- ☐ Für das Backend:

  keine Änderungen

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

  - ☐ Wurde die Datenbankenstruktur geändert?

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  nein, wurde nachgeholt ✓

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung ✓

  - ☐ Autor

- ☐ Sind alle Methoden kommentiert?

Siehe Klassen (nachgeholt)

Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

Tests

- ☐ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 45

Reviewer: Tobias

Entwickler: Ichan

Commithash: de87a0d

Datum: 14.09.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet? ✓
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art? ✓
- ☐ Sind Buttons durch ihre Position direkt ersichtlich?

  keine da
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar? ✓
- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?

  keine da
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen? ✓

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar? ✓

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt: keine Back-end Änderungen

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  X
- ☐ Gibt es obsoleten Code?

  X
- ☐ Gibt es unbenutzte Variablen?

  X
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? X

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert? keine verwendet
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

Tests

- ☐ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 47

Reviewer: Claas
Entwickler: Leo
Commithash: 3c24671 diff 5886757
Datum: 14.05.17

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
  Linienstärke wurde angemerkt und behoben
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☑ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☑ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☑ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☑ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☑ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  Linter -> wurden direkt behoben

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  nein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  nein → nächstes Refactoring?
  (wurde schnell behoben, was einfacher als gedacht

- ☐ Für das Frontend:

  - ☑ Sind alle POST Methoden richtig formatiert?

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    ~~nein → Anders~~ ja

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      ja

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

    ja

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☑ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

3

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  ja, Anmerkungen wurden eingearbeitet

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  siehe oben

- ☐ Ist die Dokumentation im Wiki vollständig?

  ja (vorbildlich ergänzt)

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  nein, wurde nachgereicht

- ☐ Wie hoch ist die Statement Coverage?

  ~~[illegible]~~. 74% wird zum Release g behoben

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 49

Reviewer: Tobias

Entwickler: Ilhan

Commithash: ~~6888796~~ de87a0d

Datum: 14.09.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet? ✓
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art? ✓
- ☐ Sind Buttons durch ihre Position direkt ersichtlich? ✓
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar? ✓
- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben? ✓
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen? ✓

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

Keine post-Requests ans Backend

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

unsinnvolle Dateinamen wurden geändert

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  nein! :)
- ☐ Gibt es obsoleten Code?

  nein
- ☐ Gibt es unbenutzte Variablen?

  nein
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? nein

US 49

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend: keine Änderungen im Backend
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

nein, wurden ergänzt

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

3

- ☐ Sind alle Methoden kommentiert?

nein, wurden ergänzt

Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung ✓
  - ☐ Autor ✓
  - ☐ Eingabeparameter ✓
  - ☐ Rückgabewert ✓

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  ✗

- ☐ Sind die Kommentare verständlich für alle Entwickler? ✓

- ☐ Ist die Dokumentation im Wiki vollständig?

braucht es nicht, also ja

Tests

- ☐ Sind Tests aller Methoden vorhanden? ✓

- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US-50

Reviewer: ilhan
Entwickler: Claas
Commithash:
Datum:

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?

  keine neuen Schaltflächen

- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☐ Sind Buttons durch ihre Position direkt ersichtlich?
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

Alles in Ordnung

- ☐ Gibt es Meldungen der Tools?
- ☐ Gibt es obsoleten Code?
- ☐ Gibt es unbenutzte Variablen?
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

- ☐ Wurden Hilfsmethoden ausgelagert? Ja
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☑ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor
  - ☑ Eingabeparameter
  - ☑ Rückgabewert

- ☑ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☑ Sind die Kommentare verständlich für alle Entwickler?
- ☑ Ist die Dokumentation im Wiki vollständig?

Tests

- ☑ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 51

Reviewer: Tobias

Entwickler: Ilhan

Commithash: de87a0d

Datum: 14.09.

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet? wird geändert.

  Nein „Durchsuchen..." nicht.

- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art? ✓

- ☐ Sind Buttons durch ihre Position direkt ersichtlich? ✓

- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?

  Ja, da keine Nebeninfos vorhanden ✓

- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben? ✓

- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen? ✓

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  (geändert)

  default wird noch nicht richtig gesetzt
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  Ja, wurden behoben
- ☐ Gibt es obsoleten Code?

  jetzt nicht mehr
- ☐ Gibt es unbenutzte Variablen?

  X
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? X

US 51

- ☐ Wurden Hilfsmethoden ausgelagert?

  ✓ (upload)

- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert? ✓
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert? ✓
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Wiki wurde editiert

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück? ✓
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert? ✓

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst: vorhandene ~~funkt.~~ Classen wurden nur erweitert Funktionen implizit

  - ☐ Beschreibung der Funktion und Verwendung ✓
  - ☐ Autor

3

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

x (wurde ergänzt)

Tests

- ☐ Sind Tests aller Methoden vorhanden?

x (wurden erweitert)

- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 54

Reviewer: Claas
Entwickler: Tobias
Commithash: 3858c4 diff 3c24671c
Datum: 13.05.2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?

  ja

- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?

  nein, wurde aber gefixed

- ☐ Sind Buttons durch ihre Position direkt ersichtlich?

  ja

- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?

  ja

- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?

  ja

- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

  ja

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar? JZ

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  gibt keinen

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

  ~~keine Änderungen~~ ja

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

  ja

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  ja

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  Ja

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  ja, wurden gefixed

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  nein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  nein

54

- ☐ Wurden Hilfsmethoden ausgelagert?

  ✓

- ☐ Für das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?

    ja

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    ✓

  - ☐ Gibt es Änderungen?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      ✓

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

    ja

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

✓

Kommentare

- ☐ Sind alle Klassen kommentiert?

  nein (Frontend fehlt) wurde nachgeliefert

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

  - ☐ Autor

- ☐ Sind alle Methoden kommentiert?

  nein (Backend fehlt) wurde nachgeliefert

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nein

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  Ja

- ☐ Ist die Dokumentation im Wiki vollständig?

  Ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  Ja

- ☐ Wie hoch ist die Statement Coverage?

  77 % (wird gefixed) vor dem Release

4

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 55

Reviewer: ~~Fab~~ Claas
Entwickler: Tobias
Commithash: 3858c4 diff 3c2467c
Datum: 13.03.2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?

  Ja

- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?

  Ja

- ☐ Sind Buttons durch ihre Position direkt ersichtlich?

  Ja

- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?

  Ja

- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?

  Ja

- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

  Ja

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar? Ja

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  ✓

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

  ✓

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

  ✓

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  ✓

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  ✓

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  wurden gefixed

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  kein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  nein

55

- ☐ Wurden Hilfsmethoden ausgelagert?

  n/a

- ☐ Für das Frontend:

  - ☐ Sind alle POST Methoden richtig formatiert?

    ✓

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    ja

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      die Änderungen wurden dokumentiert

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

    ja

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  alle wurde angemerkt & behoben

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

  - ☐ Autor

- ☐ Sind alle Methoden kommentiert?

  Wurde angemerkt und behoben

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nein

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  Ja, ausreichend

- ☐ Ist die Dokumentation im Wiki vollständig?

  Ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  Ja

- ☐ Wie hoch ist die Statement Coverage?

  77 %

Veränderbarkeit

ng lint Output

```
npm info it worked if it ends with ok
npm info using npm@5.0.0
npm info using node@v8.0.0
npm info lifecycle clonecademy@0.0.0~prelint: clonecademy@0.0.0
[..................] / : info lifecycle clonecademy@0.0.0~prelint: clonecademy@

> clonecademy@0.0.0 lint /angular
> ng lint

Warning: The 'no-use-before-declare' rule requires type checking

ERROR: /angular/src/app/learn/course/course_statistics/statistics.component.ts
    ↪ [74, 23]: trailing whitespace
ERROR: /angular/src/app/learn/course/course_statistics/statistics.component.ts
    ↪ [56, 21]: != should be !==
ERROR: /angular/src/app/learn/course/course_statistics/statistics.component.ts
    ↪ [56, 48]: != should be !==

Lint errors found in the listed files.
npm info lifecycle clonecademy@0.0.0~lint: Failed to exec lint script
npm ERR! code ELIFECYCLE
npm ERR! errno 2
npm ERR! clonecademy@0.0.0 lint: `ng lint`
npm ERR! Exit status 2
npm ERR!
npm ERR! Failed at the clonecademy@0.0.0 lint script.
npm ERR! This is probably not a problem with npm. There is likely additional
    ↪ logging output above.

npm ERR! A complete log of this run can be found in:
npm ERR! /root/.npm/_logs/2017-09-25T15_22_25_720Z-debug.log
```

pylint Output

```
************* Module learning_base.models
E: 72,21: Module 'django.utils.timezone' has no 'localdate' member (no-member)
************* Module learning_base.views
W:172, 0: TODO: Try if the conformate "ans" instead of "error" (fixme)
W:780, 0: TODO: implement proper send_mail() (fixme)
```

```
W:813, 0: TODO: fix if an localization issues arrise (fixme)
E:814,34: Module 'django.utils.timezone' has no 'localdate' member (no-member)
C: 20, 0: external import "from django.utils.crypto import get_random_string"
    ↪ should be placed before "from . import custom_permissions" (wrong-import
    ↪ -order)
C: 20, 0: Imports from package django are not grouped (ungrouped-imports)
************* Module learning_base.admin
W: 3, 0: Wildcard import models (wildcard-import)
W: 5, 0: Wildcard import info.models (wildcard-import)
W: 3, 0: Unused import models from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import sha512 from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import PolymorphicModel from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import timezone from wildcard import (unused-wildcard-import)
W: 3, 0: Unused import started_courses from wildcard import (unused-wildcard-
    ↪ import)
W: 3, 0: Unused import Question from wildcard import (unused-wildcard-import)
************* Module learning_base.info.__init__
R: 1, 0: Cyclic import (learning_base.info.models -> learning_base.info.
    ↪ serializer) (cyclic-import)

------------------------------------------------------------------
Your code has been rated at 9.83/10 (previous run: 8.57/10, +1.26)
```

coverage Output

Coverage report: 81%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	16	0	0	100%
django/learning_base/custom_permissions.py	14	2	0	86%
django/learning_base/info/__init__.py	0	0	0	100%
django/learning_base/info/models.py	42	16	0	62%
django/learning_base/info/serializer.py	23	5	0	78%
django/learning_base/models.py	150	21	0	86%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	38	3	0	92%
django/learning_base/multiple_choice/serializer.py	52	1	0	98%
django/learning_base/serializers.py	309	64	0	79%
django/learning_base/tests.py	422	5	0	99%
django/learning_base/views.py	423	170	0	60%
django/manage.py	13	6	0	54%
Total	**1534**	**293**	**0**	**81%**

coverage.py v4.4.1, created at 2017-09-27 16:30

Datensicherheit

bandit Output

```
Run started:2017-09-27 14:36:41.013624

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 3792
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.13 Iteration 19 - 21.09.

Abgegebene Userstories: 23, 29, 48, 52, 56, 57, 58

Commit Hash:

Checklisten

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 29 (nicht abgabebereit)

Reviewer: Claas
Entwickler: Tobias
Commithash: 3858c4 diff ~~d751cc4~~ 3c24671c
Datum: 13.03.2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet?
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☐ Sind Buttons durch ihre Position direkt ersichtlich?
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?
- ☐ Gibt es obsoleten Code?
- ☐ Gibt es unbenutzte Variablen?
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

Tests

- ☐ Sind Tests aller Methoden vorhanden?
- ☐ Wie hoch ist die Statement Coverage?

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 29

Reviewer: Claas
Entwickler: Tobias
Commithash: ec3be2a diff 70870ad
Datum: 21. 09. 2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

- ☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  kein Input vorhanden

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

  Ja

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

  Ja

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  Ja

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  Ja (unkritisch, da im Test)

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  nein, sehr ordentlich

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  nein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  Ja na

- ☐ Für das Frontend:

  - ☑ Sind alle POST Methoden richtig formatiert?

- ☐ Für das Backend:

  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?

    nein, Änderungen waren sinnvoll

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      Ja (Wiki)

  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?

    Ja

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

      –

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Ja

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☑ Autor

- ☐ Sind alle Methoden kommentiert?

  nein, wurde direkt behoben

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☐ Autor

    → fehlt → wurden behoben
  - ☐ Eingabeparameter

    → fehlt →
  - ☑ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  nein

- ☐ Sind die Kommentare verständlich für alle Entwickler?

  Ja

- ☐ Ist die Dokumentation im Wiki vollständig?

  Ja

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  Ja

- ☐ Wie hoch ist die Statement Coverage?

  81 %

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 48

Reviewer: Tobias Huber
Entwickler: Leonhard Wiedmann
Commithash: be 87f6b
Datum: 21.09.17

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet? ja
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
  keine Schaltflächen ✓
- ☐ Sind Buttons durch ihre Position direkt ersichtlich? ✓
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar? ✓
- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben? ✓
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen? ✓

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar? ✓

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  kein Input

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert? ✓
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)? ✓
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert? ✓

  (keine)

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden? ✓

  (keine Fehler)

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools? ✓
- ☐ Gibt es obsoleten Code? ✓
- ☐ Gibt es unbenutzte Variablen? ✓
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert? ✓

US 48

- ☐ Wurden Hilfsmethoden ausgelagert?
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert?
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert?
    - ☐ Wenn ja: Sind diese begründet und dokumentiert? wik: ~~W~~urde ergänzt
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück?
  - ☐ Wurde die Datenbankenstruktur geändert?
    - ☐ ~~Wenn ja: Sind diese begründet und dokumentiert?~~

Kommentare

- ☐ Sind alle Klassen kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☐ Sind alle Methoden kommentiert?

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung ✓
  - ☐ Autor ✓
  - ☐ Eingabeparameter ✓
  - ☐ Rückgabewert ✓

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten) ✓
- ☐ Sind die Kommentare verständlich für alle Entwickler? ✓
- ☐ Ist die Dokumentation im Wiki vollständig? ✓

Tests

- ☐ Sind Tests aller Methoden vorhanden? ✓
- ☐ Wie hoch ist die Statement Coverage? 81%

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 52

Reviewer: Leonhard Wiedmann
Entwickler: Claas Völkel
Commithash: 70870ad diff ab79fde
Datum: 21.09.2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☐ Wurden nur Elemente des Material Designs verwendet? ✓
- ☐ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art? ✓
- ☐ Sind Buttons durch ihre Position direkt ersichtlich? ✓
- ☐ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar? ✓
- ☐ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben? ✓
- ☐ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen? ✓

1

- ☐ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?
- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?
- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?
- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?
- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?
- ☐ Gibt es obsoleten Code?
- ☐ Gibt es unbenutzte Variablen?
- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

- ☐ Wurden Hilfsmethoden ausgelagert? ✓
- ☐ Für das Frontend:
  - ☐ Sind alle POST Methoden richtig formatiert? ✓
- ☐ Für das Backend:
  - ☐ Sind alle Schnittstellen wie abgesprochen implementiert? ✓
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?
      Nein, wird nachgereicht
  - ☐ Geben alle Views einen gültigen und passenden Status Code zurück? ✓
  - ☐ Wurde die Datenbankenstruktur geändert? ✗
    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert? Keine neuen Klassen

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor

- ☐ Sind alle Methoden kommentiert? Keine neuen Methoden

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung
  - ☐ Autor
  - ☐ Eingabeparameter
  - ☐ Rückgabewert

- ☐ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)

  Nein, wird nachgereicht

- ☐ Sind die Kommentare verständlich für alle Entwickler?
- ☐ Ist die Dokumentation im Wiki vollständig?

Tests

- ☐ Sind Tests aller Methoden vorhanden? ✓
- ☐ Wie hoch ist die Statement Coverage? 81%

Checklisten - Clonecademy

TECHNISCHE UNIVERSITÄT DARMSTADT

US 56

Reviewer: Claas
Entwickler: Tobias
Commithash: 2564dd2 diff cb3f2de
Datum: 21.05.2017

Checklisten

Checkliste Bedienbarkeit

Nach jeder User Story muss für die Elemente folgende Checkliste beachtet werden:

- ☑ Wurden nur Elemente des Material Designs verwendet?
- ☑ Sind Schaltflächen optisch von ihrem Hintergrund abgehoben und im gleichen Format gehalten, wie andere der selben Art?
- ☑ Sind Buttons durch ihre Position direkt ersichtlich?
- ☑ Ist wichtiger Text von Nebeninformationen abgetrennt und deutlich sichtbar?
- ☑ Sind die Schaltflächen richtig beschriftet und ist durch die Beschriftung erkennbar, welche Funktion sie haben?
- ☑ Sind neue Elemente optisch gut unterscheidbar von vorhandenen Elementen?

1

☑ Ist alles in der Mindestauflösung (1024x768) erkennbar?

Checkliste Datensicherheit

Fragen zu jedem Codeabschnitt:

- ☐ HTML Input: Wird jeglicher Input korrekt nach Cross-Site-Scripts durchsucht, bevor er als sicher markiert wird?

  keines vorhanden

- ☐ Sind die Django-Features zum Schutz vor Cross Site Request Forgery (CSRF) aktiviert?

  Ja

- ☐ Ist jede Schnittstelle mit passender Authentifizierung und Autorisierung versehen, sowie gegebenenfalls auch mit einer Anfragenbegrenzung (Throttling)?

  ja

- ☐ Werden alle Nutzereingaben nach Fehlern gefiltert?

  Ja

- ☐ Sind alle Security Probleme, die bei der softwarebasierten Analyse aufgetreten sind, behoben, oder aus dem Kontext heraus als unkritisch betrachtet worden?

  Ja (gibt keine)

Checkliste Veränderbarkeit

Code Qualität

- ☐ Gibt es Meldungen der Tools?

  nein

- ☐ Gibt es obsoleten Code?

  nein

- ☐ Gibt es unbenutzte Variablen?

  nein

- ☐ Wurden bereits vorhandene Funktionalitäten neu implementiert?

  nein

- ☐ Wurden Hilfsmethoden ausgelagert?

  n/a

- ☐ Für das Frontend:

  - ☑ Sind alle POST Methoden richtig formatiert?

- ☐ Für das Backend:

  - ☑ Sind alle Schnittstellen wie abgesprochen implementiert?

    - ☑ Wenn ja: Sind diese begründet und dokumentiert?

  - ☑ Geben alle Views einen gültigen und passenden Status Code zurück?

  - ☐ Wurde die Datenbankenstruktur geändert?

    nein

    - ☐ Wenn ja: Sind diese begründet und dokumentiert?

Kommentare

- ☐ Sind alle Klassen kommentiert?

  ja

  Dies umfasst:

  - ☐ Beschreibung der Funktion und Verwendung

  - ☐ Autor

- ☐ Sind alle Methoden kommentiert?

  nein, wurde gefixed

  Dies umfasst:

  - ☑ Beschreibung der Funktion und Verwendung
  - ☒ Autor

    fehlt
  - ☐ Eingabeparameter

    fehlt
  - ☐ Rückgabewert

    fehlt
- ☑ Gibt es weitere, schwer verständliche Stellen, die kommentiert werden sollten? (Anmerkungen der anderen Entwickler beachten)
- ☑ Sind die Kommentare verständlich für alle Entwickler?
- ☑ Ist die Dokumentation im Wiki vollständig?

Tests

- ☐ Sind Tests aller Methoden vorhanden?

  Ja
- ☐ Wie hoch ist die Statement Coverage?

  81%

Veränderbarkeit

ng lint Output

pylint Output

```
************* Module learning_base.default_picture
C: 1, 0: Line too long (17174/100) (line-too-long)
C: 2, 0: Trailing newlines (trailing-newlines)
C: 1, 0: Missing module docstring (missing-docstring)
C: 1, 0: Invalid constant name "default_picture" (invalid-name)
************* Module learning_base.views
W:845, 0: TODO: implement proper send_mail() (fixme)
W:878, 0: TODO: fix if an localization issues arrise (fixme)
C: 1, 0: Too many lines in module (1011/1000) (too-many-lines)
R:682, 4: Too many branches (23/12) (too-many-branches)
R:682, 4: Too many statements (60/50) (too-many-statements)
************* Module learning_base.serializers
R:245, 4: Too many branches (22/12) (too-many-branches)
************* Module learning_base.info.models
W: 32,17: Unused argument 'data' (unused-argument)
E: 69,41: Instance of 'InformationText' has no 'id' member (no-member)
W: 96,17: Unused argument 'data' (unused-argument)
E:136,41: Instance of 'InformationYoutube' has no 'id' member (no-member)
************* Module learning_base.multiple_choice.__init__
R: 1, 0: Cyclic import (learning_base.multiple_choice.models -> learning_base.
    ↪ multiple_choice.serializer) (cyclic-import)
R: 1, 0: Cyclic import (learning_base.info.models -> learning_base.info.
    ↪ serializer) (cyclic-import)

------------------------------------------------------------------
Your code has been rated at 9.78/10 (previous run: 9.78/10, +0.00)
```

coverage Output

Coverage report: 90%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	16	0	0	100%
django/learning_base/custom_permissions.py	14	1	0	93%
django/learning_base/default_picture.py	1	0	0	100%
django/learning_base/info/__init__.py	0	0	0	100%
django/learning_base/info/models.py	41	3	0	93%
django/learning_base/info/serializer.py	22	5	0	77%
django/learning_base/models.py	149	8	0	95%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	38	3	0	92%
django/learning_base/multiple_choice/serializer.py	51	1	0	98%
django/learning_base/serializers.py	299	32	0	89%
django/learning_base/tests.py	631	5	0	99%
django/learning_base/views.py	456	117	0	74%
django/manage.py	13	6	0	54%
Total	**1763**	**181**	**0**	**90%**

coverage.py v4.4.1, created at 2017-09-28 15:34

Datensicherheit

bandit Output

```
Run started:2017-09-27 14:36:41.013624

Test results:
       No issues identified.

Code scanned:
       Total lines of code: 3792
       Total lines skipped (#nosec): 0

Run metrics:
       Total issues (by severity):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
       Total issues (by confidence):
              Undefined: 0.0
              Low: 0.0
              Medium: 0.0
              High: 0.0
Files skipped (0):
```

C.14 Release 1 - 07.08.2017

Nutzerstudie - Protokoll

Protokol
Nutzerstudie 28.07.2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Leonhard Wiedmann
BP Clonecademy

Aufgaben

- Registrierung
- Kurs erfolgreich abschließen
- Kurs erstellen

Probleme

Hier sind die Probleme und Anmerkungen der Probanden aufgelistet. Die Zahl in den Klammern zeigt an, wie viele der Probanden es gemerkt haben oder Probleme damit hatten.

- Die Kursliste wird nicht als solche erkannt. Es wurde immer erst auf „Courses“ geklickt. Wurde auch von mehreren Nutzern kritisiert, dass es nicht klar erkennbar ist. (4/4)
- „+ add module“ ist unverständlich. (3/4)
- anstelle „submit“ und „next question“ wäre ein Button besser und übersichtlicher. (3/4)
- Wenn eine Frage erneut falsch beantwortet wird, sollte es erkennbar sein, dass es wieder falsch ist. (4/4)
- Bei ungültiger eingabe des Benutzernamens richtige Fehler anzeigen (2/4)
- Der Button „start Course“ ist verwirrend und sollte nicht extra im Module angezeigt werden. Es wurde der vorschlage gebracht ihn in die Übersichtsliste aufzunehmen. (1/4)
- Beim Feedback sollte noch „richtig“ oder „falsch“ dabei stehen. (1/4)
- Es sollte erklärt werden, wass ein Kurs und was ein Module ist.
- multipLe und nicht multipe
- Der Fragentext fällt im Fragen Modul nicht so gut auf wie der learning text und sollte anders hervorgehoben werden (2/4)
- anstelle „learning Text“ „module description“ verwenden

Zed Attack Proxy Tool - Diskussion

Alle gefundenen Fehler betreffen nur die Konfiguration des Servers, nicht das Produkt an sich. Da wir im Gespräch mit den Auftraggeber*innen vereinbart hatten, dass wir nicht dafür zuständig sind, den Server zu sichern, ist das Korrigieren diese Fehler nicht Teil unserer Aufgabe.

Dennoch wollen wir für die kommenden Releases Empfehlungen geben, wie diese Probleme verhindert werden können. Claas recherchiert dazu zu den einzelnen Fehlern und schreibt eine kurze Anleitung, wie die richtigen Einstellung lauten. Diese werden dann an die Auftraggeber*innen weiter gegeben.

Zed Attack Proxy Tool - Output

ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	1
Low	6
Informational	0

Alert Detail

Medium (Medium)	X-Frame-Options Header Not Set
Description	X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.
URL	http://localhost:4200
Method	GET
Parameter	X-Frame-Options
Instances	1
Solution	Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. ALLOW-FROM allows specific websites to frame the web page in supported web browsers).
Reference	http://blogs.msdn.com/b/ieinternals/archive/2010/03/30/combating-clickjacking-with-x-frame-options.aspx
CWE Id	16
WASC Id	15
Source ID	3

Low (Medium)	**Web Browser XSS Protection Not Enabled**
Description	Web Browser XSS Protection is not enabled, or is disabled by the configuration of the 'X-XSS-Protection' HTTP response header on the web server
URL	http://localhost:4200
Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:4200/robots.txt
Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:4200/sitemap.xml
Method	GET
Parameter	X-XSS-Protection
Instances	3
Solution	Ensure that the web browser's XSS filter is enabled, by setting the X-XSS-Protection HTTP response header to '1'.
Other information	

	The X-XSS-Protection HTTP response header allows the web server to enable or disable the web browser's XSS protection mechanism. The following values would attempt to enable it: X-XSS-Protection: 1; mode=block X-XSS-Protection: 1; report=http://www.example.com/xss The following values would disable it: X-XSS-Protection: 0 The X-XSS-Protection HTTP response header is currently supported on Internet Explorer, Chrome and Safari (WebKit). Note that this alert is only raised if the response body could potentially contain an XSS payload (with a text-based content type, with a non-zero length).
Reference	https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet https://blog.veracode.com/2014/03/guidelines-for-setting-security-headers/
CWE Id	933
WASC Id	14
Source ID	3

Low (Medium)	**X-Content-Type-Options Header Missing**
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	http://localhost:4200/main.bundle.js
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:4200
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:4200/inline.bundle.js
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:4200/styles.bundle.js
Method	GET
Parameter	X-Content-Type-Options
Instances	4
Solution	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages. If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.
Other information	This issue still applies to error type pages (401, 403, 500, etc) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.

	At "High" threshold this scanner will not alert on client or server error responses.
Reference	http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx
	https://www.owasp.org/index.php/List_of_useful_HTTP_headers
CWE Id	16
WASC Id	15
Source ID	3

Low (Medium)	**X-Content-Type-Options Header Missing**
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	http://localhost:8000/static/admin/css/login.css
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:8000/admin
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:8000/admin/login/?next=/admin/
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:8000/static/admin/css/base.css
Method	GET
Parameter	X-Content-Type-Options
Instances	4
Solution	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.
	If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.
Other information	This issue still applies to error type pages (401, 403, 500, etc) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.
	At "High" threshold this scanner will not alert on client or server error responses.
Reference	http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx
	https://www.owasp.org/index.php/List_of_useful_HTTP_headers
CWE Id	16
WASC Id	15
Source ID	3

Low (Medium)	**Web Browser XSS Protection Not Enabled**
Description	Web Browser XSS Protection is not enabled, or is disabled by the configuration of the 'X-XSS-Protection' HTTP response header on the web server
URL	http://localhost:8000/sitemap.xml

Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:8000/admin
Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:8000/robots.txt
Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:8000/admin/login/?next=/admin/
Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:8000/admin/login/?next=/admin/
Method	POST
Parameter	X-XSS-Protection
Instances	5
Solution	Ensure that the web browser's XSS filter is enabled, by setting the X-XSS-Protection HTTP response header to '1'.
Other information	The X-XSS-Protection HTTP response header allows the web server to enable or disable the web browser's XSS protection mechanism. The following values would attempt to enable it: X-XSS-Protection: 1; mode=block X-XSS-Protection: 1; report=http://www.example.com/xss The following values would disable it: X-XSS-Protection: 0 The X-XSS-Protection HTTP response header is currently supported on Internet Explorer, Chrome and Safari (WebKit). Note that this alert is only raised if the response body could potentially contain an XSS payload (with a text-based content type, with a non-zero length).
Reference	https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet https://blog.veracode.com/2014/03/guidelines-for-setting-security-headers/
CWE Id	933
WASC Id	14
Source ID	3

Low (Medium)	**Cookie No HttpOnly Flag**
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
URL	http://localhost:8000/admin/login/?next=/admin/
Method	GET

Parameter	csrftoken
Evidence	Set-Cookie: csrftoken
URL	http://localhost:8000/admin
Method	GET
Parameter	csrftoken
Evidence	Set-Cookie: csrftoken
Instances	2
Solution	Ensure that the HttpOnly flag is set for all cookies.
Reference	http://www.owasp.org/index.php/HttpOnly
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	**Password Autocomplete in Browser**
Description	The AUTOCOMPLETE attribute is not disabled on an HTML FORM/INPUT element containing password type input. Passwords may be stored in browsers and retrieved.
URL	http://localhost:8000/admin/login/?next=/admin/
Method	GET
Parameter	id_password
Evidence	<input type="password" name="password" required id="id_password" />
URL	http://localhost:8000/admin
Method	GET
Parameter	id_password
Evidence	<input type="password" name="password" required id="id_password" />
Instances	2
Solution	Turn off the AUTOCOMPLETE attribute in forms or individual input elements containing password inputs by using AUTOCOMPLETE='OFF'.
Reference	http://www.w3schools.com/tags/att_input_autocomplete.asp https://msdn.microsoft.com/en-us/library/ms533486%28v=vs.85%29.aspx
CWE Id	525
WASC Id	15
Source ID	3

C.15 Release 2 - 13.09.2017

Nutzerstudie - Protokoll

Protokol
Nutzerstudie 06.09.2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Leonhard Wiedmann
BP Clonecademy

Aufgaben

- Registrierung
- Kurs erfolgreich abschließen
- Passwort ändern

Probleme

Hier sind die Probleme und Anmerkungen der Probanden aufgelistet. Die Zahl in den Klammern zeigt an, wie viele der Probanden es gemerkt haben oder Probleme damit hatten. Die Zahl in [] gibt an welcher issue id es in Github hat.

- nach erfolgreichen ändern des Passwort fehlt ein feedback (6/6) [40]
- submit gibt kein feedback, weder für richtig oder Falsch (4/6) [38]
- unklar wie man sich in dem Fragen Fenster bewegt und Aufgaben abgibt oder weiter kommt (1/6) [43]
- Rechtschreibfehler: "Login with you accountünd im register beim zweiter Passwort. Wurde direkt gefixt
- Üser Details"wurde als Menüpunkt kritisiert, da es nicht sofort klar ist, was damit gemeint ist (1/6) Eine Nutzerin, welche ihrer eigenen aussage nach wenig Erfahrung mit Computer und Onlineplatformen hat, hat es sofort gefunden.
- Klarere Unterscheidung von Dashboard zu Question. Der Nutzer hat kritisiert, dass es unklar ist, ob man sich noch auf dem Dashboard oder der Fragenseite ist. (1/6)
- Buttons für Kurse im Dashboard Menü sind nicht ersichtlich. (Wenn man auf Course klickt, hätte der Nutzer gerne ein extra feedback. Kann aber auch dadurch gelöst werden indem mehrere Kurse in einem Reiter stehen, wodurch es vielleicht ersichtlicher wird.) (1/6)

Der in der letzten Iteration gefundene Fehler mit einer Gefährlichkeit von ”Medium” wurde inzwischen behoben. Die verbleibenden Fehler wurden bislang nicht bearbeitet. Claas hat die notwendigen Daten allerdings an die Auftraggeber*innen geschrieben.

Wir empfehlen erneut, die Serverkonfiguration zu ändern.

Anhängend ein Auszug aus der beispielhaften Server-Konfiguration:

```
# nginx-app.conf

# the upstream component nginx needs to connect to
upstream django {
    server unix:/home/docker/django/app.sock; # for a file socket
    # server 127.0.0.1:8001; # for a web port socket (we'll use this first)
}

# configuration of the server
server {
    # the port your site will be served on, default_server indicates that this
        ↪ server block
    # is the block to use if no blocks match the server_name
    listen 80 default_server;

    # the domain name it will serve for
    server_name .example.com; # substitute your machine's IP address or FQDN
    charset utf-8;

    # max upload size
    client_max_body_size 75M; # adjust to taste

    # Recommended security headers
    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-Content-Type-Options nosniff;
    add_header X-XSS-Protection "1; mode=block";



    # Django media
    location /media {
        alias /home/docker/persistent/media; # your Django project's media files
            ↪  - amend as required
    }
```

```
    location /static {
        alias /home/docker/volatile/static; # your Django project's static files
            ↪  - amend as required
    }

    location /api {
        uwsgi_pass django;
        include /home/docker/django/uwsgi_params; # the uwsgi_params file you
            ↪ installed
    }
    # Finally, send all non-media requests to the Django server.
    location / {
        root /home/docker/angular/;
    }

}
```

Zed Attack Proxy Tool - Output

ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	0
Low	6
Informational	0

Alert Detail

Low (Medium)	**X-Content-Type-Options Header Missing**
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	http://localhost:4200/
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:4200/main.bundle.js
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:4200
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:4200/inline.bundle.js
Method	GET
Parameter	X-Content-Type-Options
URL	http://localhost:4200/styles.bundle.js
Method	GET
Parameter	X-Content-Type-Options
Instances	5
Solution	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages. If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.
Other information	This issue still applies to error type pages (401, 403, 500, etc) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scanner will not alert on client or server error responses.
Reference	http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx

	https://www.owasp.org/index.php/List_of_useful_HTTP_headers
CWE Id	16
WASC Id	15
Source ID	3

Low (Medium)	**Web Browser XSS Protection Not Enabled**
Description	Web Browser XSS Protection is not enabled, or is disabled by the configuration of the 'X-XSS-Protection' HTTP response header on the web server
URL	http://localhost:4200
Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:4200/robots.txt
Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:4200/sitemap.xml
Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:4200/
Method	GET
Parameter	X-XSS-Protection
Instances	4
Solution	Ensure that the web browser's XSS filter is enabled, by setting the X-XSS-Protection HTTP response header to '1'.
Other information	The X-XSS-Protection HTTP response header allows the web server to enable or disable the web browser's XSS protection mechanism. The following values would attempt to enable it: X-XSS-Protection: 1; mode=block X-XSS-Protection: 1; report=http://www.example.com/xss The following values would disable it: X-XSS-Protection: 0 The X-XSS-Protection HTTP response header is currently supported on Internet Explorer, Chrome and Safari (WebKit). Note that this alert is only raised if the response body could potentially contain an XSS payload (with a text-based content type, with a non-zero length).
Reference	https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet https://blog.veracode.com/2014/03/guidelines-for-setting-security-headers/
CWE Id	933
WASC Id	14
Source ID	3

Low (Medium)	**X-Content-Type-Options Header Missing**
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early

2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

URL http://localhost:8000/static/admin/css/login.css

Method GET

Parameter X-Content-Type-Options

URL http://localhost:8000/admin

Method GET

Parameter X-Content-Type-Options

URL http://localhost:8000/admin/login/?next=/admin/

Method GET

Parameter X-Content-Type-Options

URL http://localhost:8000/static/admin/css/base.css

Method GET

Parameter X-Content-Type-Options

Instances 4

Solution Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.

If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

Other information This issue still applies to error type pages (401, 403, 500, etc) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.

At "High" threshold this scanner will not alert on client or server error responses.

Reference http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx

https://www.owasp.org/index.php/List_of_useful_HTTP_headers

CWE Id 16

WASC Id 15

Source ID 3

Low (Medium) **Web Browser XSS Protection Not Enabled**

Description Web Browser XSS Protection is not enabled, or is disabled by the configuration of the 'X-XSS-Protection' HTTP response header on the web server

URL http://localhost:8000/sitemap.xml

Method GET

Parameter X-XSS-Protection

URL http://localhost:8000/admin

Method GET

Parameter X-XSS-Protection

URL http://localhost:8000/robots.txt

Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:8000/admin/login/?next=/admin/
Method	GET
Parameter	X-XSS-Protection
URL	http://localhost:8000/admin/login/?next=/admin/
Method	POST
Parameter	X-XSS-Protection
Instances	5
Solution	Ensure that the web browser's XSS filter is enabled, by setting the X-XSS-Protection HTTP response header to '1'.
Other information	The X-XSS-Protection HTTP response header allows the web server to enable or disable the web browser's XSS protection mechanism. The following values would attempt to enable it: X-XSS-Protection: 1; mode=block X-XSS-Protection: 1; report=http://www.example.com/xss The following values would disable it: X-XSS-Protection: 0 The X-XSS-Protection HTTP response header is currently supported on Internet Explorer, Chrome and Safari (WebKit). Note that this alert is only raised if the response body could potentially contain an XSS payload (with a text-based content type, with a non-zero length).
Reference	https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet https://blog.veracode.com/2014/03/guidelines-for-setting-security-headers/
CWE Id	933
WASC Id	14
Source ID	3

Low (Medium)	**Cookie No HttpOnly Flag**
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
URL	http://localhost:8000/admin/login/?next=/admin/
Method	GET
Parameter	csrftoken
Evidence	Set-Cookie: csrftoken
URL	http://localhost:8000/admin
Method	GET
Parameter	csrftoken
Evidence	Set-Cookie: csrftoken

Instances	2
Solution	Ensure that the HttpOnly flag is set for all cookies.
Reference	http://www.owasp.org/index.php/HttpOnly
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	**Password Autocomplete in Browser**
Description	The AUTOCOMPLETE attribute is not disabled on an HTML FORM/INPUT element containing password type input. Passwords may be stored in browsers and retrieved.
URL	http://localhost:8000/admin/login/?next=/admin/
Method	GET
Parameter	id_password
Evidence	<input type="password" name="password" required id="id_password" />
URL	http://localhost:8000/admin
Method	GET
Parameter	id_password
Evidence	<input type="password" name="password" required id="id_password" />
Instances	2
Solution	Turn off the AUTOCOMPLETE attribute in forms or individual input elements containing password inputs by using AUTOCOMPLETE='OFF'.
Reference	http://www.w3schools.com/tags/att_input_autocomplete.asp https://msdn.microsoft.com/en-us/library/ms533486%28v=vs.85%29.aspx
CWE Id	525
WASC Id	15
Source ID	3

C.16 Release 3 - 01.10.2017

Nutzerstudie - Protokoll

Protokol
Nutzerstudie 06.09.2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Leonhard Wiedmann
BP Clonecademy

Aufgaben

- Registrierung
- Kurs mit Quiz erfolgreich abschließen
- Passwort ändern

Probanden

Es haben 3 Männer und Frauen an der Nutzerstudie teilgenommen.

Probleme

Hier sind die Probleme und Anmerkungen der Probanden aufgelistet. Die Zahl in den Klammern zeigt an, wie viele der Probanden ähnliche Probleme hatten. Die Zahl in

gibt an welcher Issue in Github daraus resultiert.

- Register mit einer Ungültigen Email gibt keinen Fehler.
- Deutlicherer ”Submit” Button, so wie ”next Question” ist verwirrend
- Der Übergang vom Kurs zum Quiz ist nicht erkennbar.
- Das Ende (Feedback) vom Quiz fehlt.
- der Text ”Your answer is correct/not correct” im Quiz ist bei langen Fragen nicht sofort erkennbar.
- View Profile führt direkt zu Settings, was alle Nutzer erst mal verwirrt hat und sich erst nicht sicher waren ob sie richtig sind um das Passwort zu ändern.
- Der Passwort Submit Text sollte lauten "Please enter your old passwordöder so ähnlich, da Nutzer dort nochmal ihr neues Passwort eingeben wollten.
- Durch ändern des Passwort wird das Bild des Nutzers gelöscht.

63

-

D Progression des Design

Planungsphase

Dies war der ursprüngliche Plan für das Layout der Seite. Dieses Design wurde mit den Auftraggeber*innen erarbeitet.

Minimum Viable Product

Im folgenden ist ein Screenshot vom Frontend gezeigt, nachdem die Webseite funktional war.

Abgabefertiges Produkt

Im folgenden ist der Stand des Produktes bei Abgabe zu sehen.

CloneCademy
Admin
Learn
iliricon
Is this real life?
Cheesecake!
Realitii?
Reality is unknowable!
Is this just fantasy?
Your answer was correct
Next

CloneCademy
Admin
Learn
iliricon
This is the ultimate testing course
This is the ULTIMATE course, with videos, blackjack and hookers.
Module 1: Videos and Texts
"Les Miserables sont miserable"
What do you need to know about "Les miserables"?
Main characters (1)
Main characters (2)
Module 2: Paradise Lost
This is an epic poem
Text excerpt
Writer
Jean Valjean
Inspector Javert
Fantime
submit

E Dokumentation des Codes - Auszüge

Wir haben uns dafür entschieden, jeweils drei Klassen aus dem Front- und dem Backend vorzuweisen. Dies liegt daran, dass wir die Komponenten separat entwickelt haben und die Qualität in beiden Teilen des Projektes wichtig für die weitere Verwendbarkeit ist.

Course Service - Frontend

Der Course Service ist der zentrale Dreh- und Angelpunkt über den das Frontend die Lerndaten vom Backend anfragt.

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

import {ServerService} from './server.service'
import {UserService} from './user.service';

@Injectable()
/**
 * A service to get all course related data
 *
 * @author Leonhard Wiedmann
 */
export class CourseService {

  // the courses
  data = {};
  // categories to filter the courses
  categorys: any;

  constructor(private server: ServerService, private user: UserService) {
  }

  /**
   load courses for all categories for the current user language
   after loading sorts course for every category in data variable
   **/
  load() {
    return new Promise((resolve, reject) => {
      this.getCategory().then(() => {
        let requests = 0
        for (let i = 0; i < this.categorys.length; i++) {
          this.server.post('courses/', {
            'type': '',
```

```
        'category': this.categorys[i].name,
        'language': this.user.language
      }, true)
        .then((data) => {
           requests++;
           this.data[this.categorys[i].name] = data;
           if (requests === this.categorys.length) {
             resolve()
           }
          }
        )
        .catch(err => {
          reject(err)
        })
     }
    })
  })
}

/**
 get all categories from server
 will store all categories in the categories variable as array
 @author Leonhard Wiedmann
 @returns void
 **/
getCategory() {
  return new Promise((resolve, reject) => this.server.get('get-course-
     ↪ categories/', true)
    .then(data => {
       this.categorys = data;
       resolve(data);
     }
    )
    .catch(err => {
     reject(err);
    }))
}

/**
 returns promise and resolve if this course exists.
 Loads the courses if currently not loaded.
 @author Leonhard Wiedmann
 @returns a promise resolving the course
 @param id the id of the course loaded
 **/
contains(id: number) {
```

```
  return new Promise((resolve, reject) => {
    // loads the course if the data is not in cache
    if (this.data == null) {
      this.load().then(() => {
          const value = this.get(id);
          if (value) {
            resolve(value);
          } else {
            reject();
          }
        }
      )
        .catch(() => {
          reject()
        })
    } else {
      const value = this.get(id);
      if (value) {
        resolve(value);
      } else {
        reject();
      }
    }
  })
}

/**
 Returns the course by id
 @author Leonhard Wiedmann
 @aram id
 @returns a loaded course
 **/
get(id: number) {
  for (let i = 0; i < this.categorys.length; i++) {
    for (let j = 0; j < this.data[this.categorys[i].name].length; j++) {
      if (this.data[this.categorys[i].name][j]['id'] === Number(id)) {
        return this.data[this.categorys[i].name][j];
      }

    }
  }

  return false;
}

/**
```

```
   Returns all courses that a user has started already
   @author Claas Voelcker
   @returns all started courses by the user
   **/
  get_started(): {}[] {
    const courses = [];
    for (let i = 0; i < this.categorys.length; i++) {
      for (let j = 0; j < this.data[this.categorys[i].name].length; j++) {
        // checks if more then one and less then all questions were answered
        const num_answered = this.data[this.categorys[i].name][j]['num_answered']
            ↪ ];
        const num_total = this.data[this.categorys[i].name][j]['num_questions'];
        if (num_answered > 0 && num_answered < num_total) {
          courses.push(this.data[this.categorys[i].name][j]);
        }
      }
    }
    return courses
  }

}
```

User Details - Frontend

User Details ist eine komplexe Klasse, die es ermöglicht, Einstellungen zu verändern. Sie zeigt gut, wie der Code des Frontends aufgebaut ist.

```
import {Component, Input} from '@angular/core';
import {ActivatedRoute, Params, Router} from '@angular/router'
import {ServerService} from '../../service/server.service';
import {UserService} from '../../service/user.service';
import {ProfilesComponent} from '../profiles/profiles.component'

/**
Make the user details visible to the admin and add some more admin funktions
    ↪ for users
@author Ilhan Simsiki
**/
@Component({
  selector: 'app-user-detail',
  templateUrl: './user-detail.component.html',
  styleUrls: ['./user-detail.component.sass']
})

export class UserDetailComponent {
```

```
// {username: string, id: number, email: string, group: {}, dateRegistered:
    ↪ Date, dateString: string}
user: any;
id: number;
isMod = false;
isAdmin = false;

loading = true;

position = 'before';

constructor(
  private route: ActivatedRoute,
  private server: ServerService,
  private router: Router
) {
  this.route.params.subscribe(data => {
    this.id = data.id
    this.change(this.id);
  })
}

/**
  This funktions loads the user with the id, gives the variable "user" the
  response and sets the variables isMod, isAdmin

  @input id: the id number of the user to load from the server
  @author Ilhan Simisiki
**/
change(id: number) {
  this.loading = true;
  // load the current user
  this.server.get('user/' + id + '/')
    .then(data => {
      this.user = data
      this.user['dateRegistered'] = new Date(data['date_joined'])
      this.isMod = (-1 !== this.user['groups'].indexOf('moderator'));
      this.isAdmin = (-1 !== this.user['groups'].indexOf('admin'));

      // show the spinning loader until the user is loaded
      this.loading = false;
    })
}

/**
```

```
  Promote or demote a user to admin or moderator and reset the current user
     ↪ information

  @input
    right: a string for the group ('admin' or 'moderator')
    action: a string for 'demote' or 'promote'
  @author Tobias Huber
  **/
  proDemote(right: string, action: string) {
    this.server.post('user/' + this.id + '/rights', {
      'right': right,
      'action': action
    })
    .then(data => {
      this.isMod = (-1 !== data['groups'].indexOf('moderator'))
      this.isAdmin = (-1 !== data['groups'].indexOf('admin'))
    })
  }
}
```

Statistics - Frontend

Statistics zeigt den Nutzer*innen ihre Nutzungsstatistiken auf der Plattform an.

```
import {Component, OnInit, Input, ChangeDetectorRef} from '@angular/core';

import {ActivatedRoute, Params, Router} from '@angular/router'

import {ServerService} from '../../../service/server.service'

import 'rxjs/Rx' ;

/**
 * @author Leonhard Wiedmann
 *
 * A component to display the statistics of the current course
 */
@Component({
  selector: 'app-course-statistics',
  templateUrl: './statistics.component.html',
  styleUrls: ['./statistics.component.scss']
})
export class CourseStatisticsComponent implements OnInit {

  id: number;
```

```
list: any;

// Pie variables
loadingPie = false;
pieChartLabels = ['Solved', 'Not solved'];
pieChartData = [];
pieChartColor = [{ backgroundColor: [ '#aaff80' , 'darkred' ]}];

constructor(
  private server: ServerService,
  private route: ActivatedRoute,
  private cdr: ChangeDetectorRef
) {

}

ngOnInit() {
  this.route.params.subscribe(data => {
    this.id = data.id
  })
  this.loadPie()
  this.loadList()
}

/**
Load the variables for the pie view
@author Leonhard Wiedmann
**/
loadPie() {
  this.loadingPie = true;
  this.server.post('statistics', {
    course: this.id,
    filter: 'solved'
  }).then((data: any) => {
    if (data.True !== undefined && data.False !== undefined) {
      this.pieChartData = [data['True'], data['False']]
      this.loadingPie = true;
    }
    this.cdr.detectChanges()
    })
}

/**
Load the list of questions, how many tries this question has and how many
   ↪ correct tries it has
@author Leonhard Wiedmann
```

```
  **/
  loadList() {
    this.loadingPie = true;
    this.server.post('statistics', {
      list_questions: true,
      course: this.id,
    }).then((data: any) => {
      this.list = data
      this.cdr.detectChanges()
      })
  }

  /**
  Download the statistics for the current user
  @author Leonhard Wiedmann
  **/
  downloadStatistics() {
    this.server.downloadStatistics({id: 0})
  }

}
```

Course object - Backend

Das Course Objekt repräsentiert den betreffenden Eintrag in der Datenbank und stellt weitere Funktionalitäten für das Backend bereit.

```
class Course(models.Model):
    """
    One course is a group of questions which build on each other and should be
    solved together. These questions should have similar topics, difficulty
    and should form a compete unit for learning.
    :author: Claas Voelcker
    """

    class Meta:
        unique_together = ['category', 'name']

    # difficulty selection and mapping to human readable names
    EASY = 0
    MODERATE = 1
    DIFFICULT = 2
    EXPERT = 3
    DIFFICULTY = (
        (EASY, 'Easy (high school students)'),
```

```
    (MODERATE, 'Moderate (college entry)'),
    (DIFFICULT, 'Difficult (college students'),
    (EXPERT, 'Expert (college graduates)')
)

# language selection and mapping
GER = 'de'
ENG = 'en'
LANGUAGES = (
    (GER, 'German/Deutsch'),
    (ENG, 'English')
)

# the name of the course
name = models.CharField(
    verbose_name='Course name',
    help_text="A short concise name for the course",
    max_length=144
)

# foreign key mapping to the CourseCategory object
category = models.ForeignKey(
    CourseCategory,
    null=True,
    blank=True
)

# choice field mapped to dictionary above
difficulty = models.IntegerField(
    verbose_name='Course difficulty',
    choices=DIFFICULTY,
    default=MODERATE
)

# choice field mapped to dictionary above
language = models.CharField(
    verbose_name='Course Language',
    max_length=2,
    choices=LANGUAGES,
    default=ENG
)

# foreign key mapping to the user who can edit the course
responsible_mod = models.ForeignKey(
    User,
    on_delete=models.SET_NULL,
```

```
        null=True,
        blank=True
    )

    # should the course be serialized for normal users
    is_visible = models.BooleanField(
        verbose_name='Is the course visible',
        default=False
    )

    # a short description of the course
    description = models.CharField(
        max_length=144,
        null=True,
        blank=True,
        default=""
    )

    def __str__(self):
        return self.name

    def num_of_modules(self):
        """
        Returns the number of modules
        """
        return len(Module.objects.filter(course=self))
```

Question object - Backend

```
class Question(PolymorphicModel):
    """
    A question is the smallest unit of the learning process. A question has a
    task that can be solved by a user, a correct solution to evaluate the
    answer and a way to provide feedback to the user.

    :author: Claas Voelcker
    """

    class Meta:
        unique_together = ['module', 'order']
        ordering = ['module', 'order']

    # a title for the question
    title = models.TextField(
        verbose_name='Question title',
```

```
    help_text="A short and concise name for the question",
    blank=True,
    null=True
)

# the question text, that provides additional information to the user
text = models.TextField(
    verbose_name='Question text',
    help_text="This field can contain markdown syntax"
)

# the specific question
question = models.TextField(
    verbose_name='Question',
    help_text="This field can contain markdown syntax",
    blank=True,
    null=True
)

# a custom feedback that can be displayed
feedback = models.TextField(
    verbose_name="feedback",
    help_text="The feedback for the user after a sucessful answer",
    blank=True,
    null=True
)

# the ordering attribute of the question (needs to be explicitly saved)
order = models.IntegerField()

# foreign key mapping to the module that contains this question
module = models.ForeignKey(
    Module,
    verbose_name="Module",
    help_text="The corresponding module for the question",
    on_delete=models.CASCADE
)

def is_first_question(self):
    """
    Checks whether this is the first question in the module

    :author: Claas Voelcker
    :return: whether this is the first question or not
    """
    questions = self.module.question_set
```

```
        return self == questions.first()

    def is_last_question(self):
        """
        Checks whether this is the last question in the module

        :author: Claas Voelcker
        :return: whether this is the last question or not
        """
        questions = self.module.question_set
        return self == questions.last()

    def get_previous_in_order(self):
        """
        Returns the previous question in the course
        :author: Claas Voelcker
        :return: the previous question in the same module
        """
        questions = self.module.question_set.all()
        if list(questions).index(self) <= 0:
            return False
        return questions[list(questions).index(self) - 1]

    def get_points(self):
        """
        Returns the number of ranking points for the question.
        This method needs to be overridden by subclasses and
        remains unimplemented here.
        :author: Claas Voelcker
        :return: the points
        :raise: not implemented error
        """
        raise NotImplementedError

    def __str__(self):
        return self.title
```

CourseView object - Backend

Die Klasse CourseView modelliert die REST-Schnittstelle über die auf Kurse zugegriffen werden kann.

```
class CourseView(APIView):
    """
    Contains all code related to viewing and saving courses.
```

```
    :author: Claas Voelcker
    """
    authentication_classes = (authentication.TokenAuthentication,)
    permission_classes = (
        custom_permissions.IsModOrAdminOrReadOnly,)

    def get(self, request, course_id=None, format=None):
        """
        Returns a course if the course_id exists. The course, it's
        modules and questions are serialized.

        :author: Claas Voelcker
        :param request: request object containing auth token and user id
        :param course_id: the id of the required course
        :param format: unused (inherited)
        :return: a response containing the course serialization
        """

        # if no course id is given, the method was called wrong
        if not course_id:
            return Response({'ans': 'Method not allowed'},
                            status=status.HTTP_405_METHOD_NOT_ALLOWED)
        try:
            # fetch the course object, serialize it and return
            # the serialization
            course = Course.objects.get(id=course_id)
            course_serializer = serializers.CourseSerializer(course, context={
                'request': request})
            return Response(course_serializer.data,
                            status=status.HTTP_200_OK)
        # in case of an exception, throw a "Course not found" error for the
        # frontend, packaged in a valid response with an error status code
        except Exception:
            return Response({'ans': 'Course not found'},
                            status=status.HTTP_404_NOT_FOUND)

    def post(self, request, course_id=None, format=None):
        """
        Saves a course to the database. If the course id is provided,
        the method updates and existing course, otherwise, a new course
        is created.

        :author: Tobias Huber, Claas Voelcker
        :param request: request containig the user and auth token
        :param course_id: optional: the course id
                  (if a course is edited instead of created)
```

```
        :param format: unused (inherited)
        :return: a status response giving feedback about errors or a sucessful
                 database access to the frontend
        """

        data = request.data

        # checks whether the request contains any data
        if data is None:
            return Response({'error': 'Request does not contain data'},
                            status=status.HTTP_400_BAD_REQUEST)

        course_id = data.get('id')
        # Checks whether the name of the new course is unique
        if (course_id is None) and Course.objects.filter(
                name=data['name']).exists():
            return Response({'error': 'Course with that name exists'},
                            status=status.HTTP_409_CONFLICT)

        # adds the user of the request to the data
        if course_id is None:
            data['responsible_mod'] = request.user
        # if the course is edited, check for editing permission
        else:
            responsible_mod = Course.objects.get(id=course_id).responsible_mod
            # decline access if user is neither admin nor the responsible mod
            if (request.user.profile.is_admin()
                    or request.user == responsible_mod):
                data['responsible_mod'] = responsible_mod
            else:
                raise PermissionDenied(detail="You're not allowed to edit this"
                                       + "course, since you're not the"
                                       + 'responsible mod',
                                       code=None)

        # serialize the course
        course_serializer = serializers.CourseSerializer(data=data)

        # check for serialization errors
        if not course_serializer.is_valid():
            return Response({'error': course_serializer.errors},
                            status=status.HTTP_400_BAD_REQUEST)

        # send the data to the frontend
        else:
            try:
```

```
            course_serializer.create(data)
            return Response({'success': 'Course saved'},
                            status=status.HTTP_201_CREATED)
        except ParseError as error:
            return Response({'error': str(error)},
                            status=status.HTTP_400_BAD_REQUEST)
```

F Testklassen und Coverage des finalen Produkts

Dies sind die Testklassen bei Stand der Abgabe

F.1 Python Testklasse

```
from django.test import TestCase
from django.utils import timezone
from rest_framework.exceptions import ParseError

from rest_framework.test import APIRequestFactory
from rest_framework.test import force_authenticate

from django.contrib.auth.models import User, Group
from learning_base import views, models, serializers
from learning_base.models import Profile
import learning_base.multiple_choice as MultipleChoice
import learning_base.info as InformationText


class DatabaseMixin():
    def setup_database(self):
        self.factory = APIRequestFactory()

        self.admin_group = Group.objects.create(name='admin')

        self.u1 = User(username='admin')
        self.u1.save()
        self.u1.groups.add(self.admin_group)
        self.u1_profile = Profile.objects.create(user=self.u1)
        self.u1.save()

        self.category = models.CourseCategory(name='test')
        self.category.save()

        self.c1_test_en = models.Course(name='test_1', category=self.category,
                                        difficulty=0, language='en',
                                        responsible_mod=self.u1,
                                        is_visible=True)
        self.c1_test_en.save()

        self.m1_test = models.Module(name='module_1', course=self.c1_test_en,
```

```
                                order=1)
        self.m1_test.save()

        self.q1_test = MultipleChoice.models.MultipleChoiceQuestion(
            title='',
            text='a question',
            feedback='',
            order=1,
            module=self.m1_test)
        self.q1_test.save()

        self.a1_test = MultipleChoice.models.MultipleChoiceAnswer(
            question=self.q1_test,
            text='something',
            is_correct=False
        )
        self.a1_test.save()

        self.a2_test = MultipleChoice.models.MultipleChoiceAnswer(
            question=self.q1_test,
            text='something',
            is_correct=True)
        self.a2_test.save()

        self.q2_test = InformationText.models.InformationText(
            title='',
            text='an information text',
            feedback='',
            order=2,
            module=self.m1_test)
        self.q2_test.save()


class AnswerViewTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.view = views.AnswerView.as_view()
        self.setup_database()

    def test_get(self):
        request = self.factory.get('/courses/1/1/1/answers')
        force_authenticate(request, self.u1)
        response = self.view(request, 1, 0, 0)

        answer_1_serialized = serializers.get_answer_serializer(self.a1_test)
        answer_2_serialized = serializers.get_answer_serializer(self.a2_test)
```

```
        self.assertEqual(response.data,
                         [answer_1_serialized, answer_2_serialized])


class MultiCourseViewTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.factory = APIRequestFactory()
        self.view = views.MultiCourseView.as_view()

        self.setup_database()

    def test_get(self):
        request = self.factory.get('/courses/')
        force_authenticate(request, self.u1)
        response = self.view(request)
        self.assertEqual(response.status_code, 405)

    def test_post(self):
        request_1 = self.factory.post('/courses/', {'type': '',
                                                    'language': 'de',
                                                    'category': ''})
        request_1.user = self.u1

        request_2 = self.factory.post('/courses/', {'type': '',
                                                    'language': 'en',
                                                    'category': ''})
        request_3 = self.factory.post('/courses/',
                                      {'stuff': 'kajiger'})

        force_authenticate(request_1, self.u1)
        force_authenticate(request_2, self.u1)
        force_authenticate(request_3, self.u1)

        c1_test_en_serialized = serializers.CourseSerializer(
            self.c1_test_en, context={'request': request_1}).data

        response_1 = self.view(request_1)
        response_2 = self.view(request_2)
        response_3 = self.view(request_3)

        self.assertEqual(response_1.status_code, 200)
        self.assertEqual(response_1.data, [])

        self.assertEqual(response_2.status_code, 200)
        self.assertEqual(response_2.data, [c1_test_en_serialized])
```

```
        self.assertEqual(response_3.status_code, 400)


class CourseViewTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.factory = APIRequestFactory()
        self.view = views.CourseView.as_view()

        self.setup_database()

    def test_get(self):
        request = self.factory.get('/courses/1')
        request.user = self.u1
        force_authenticate(request, self.u1)

        c1_test_en_serialized = serializers.CourseSerializer(
            self.c1_test_en, context={'request': request}).data

        response = self.view(request, course_id=1)
        self.assertEqual(response.status_code, 200)
        self.assertEqual(response.data, c1_test_en_serialized)

    def test_post(self):
        request = self.factory.post('/courses/save',
                                    {'name': 'test_2', [...],
                                        'language': 'en'}, format='json')
        force_authenticate(request, self.u1)
        response = self.view(request)
        self.assertEqual(response.status_code, 201)
        self.assertTrue(models.Course.objects.filter(name='test_2').exists())

        request = self.factory.post('/courses/save',
                                    {'name': 'test_2', [...],
                                        'language': 'en'}, format='json')
        force_authenticate(request, self.u1)
        response = self.view(request)
        self.assertEqual(response.status_code, 409)

        request = self.factory.post('/courses/save',
                                    {'name': 'test_3', [...],
                                        'language': 'en'}, format='json')
        force_authenticate(request, self.u1)
        response = self.view(request)
        self.assertEqual(response.status_code, 201)
        self.assertTrue(models.Course.objects.filter(name='test_3').exists())
        self.assertTrue(models.Module.objects.filter(name='a module').exists())
```

```
        self.assertTrue(
            models.Module.objects.filter(name='another module').exists())

        request = self.factory.post('/courses/save',
                                    {'name': 'test_4', [...],
                                        'language': 'en'}, format='json')
        force_authenticate(request, self.u1)
        response = self.view(request)
        self.assertEquals(response.status_code, 400)
        self.assertFalse(models.Course.objects.filter(name='test_4').exists())
        self.assertFalse(
            MultipleChoice.models.MultipleChoiceQuestion.objects.filter(
                title='any module').exists())

        request = self.factory.post('/courses/save',
                                    {'name': 'test_4', [...],
                                        'language': 'en'}, format='json')
        force_authenticate(request, self.u1)
        response = self.view(request)
        self.assertEqual(response.status_code, 201)
        self.assertTrue(models.Course.objects.filter(name='test_4').exists())
        self.assertTrue(
            MultipleChoice.models.MultipleChoiceQuestion.objects.filter(
                title='a question').exists())

    def test_information_text(self):
        request = self.factory.post('/courses/save',
                                    {'name': 'test_4', [...],
                                        'language': 'en'}, format='json')
        force_authenticate(request, self.u1)
        response = self.view(request)
        self.assertEqual(response.status_code, 201)
        self.assertTrue(models.Course.objects.filter(name='test_4').exists())
        self.assertTrue(
            InformationText.models.InformationText.objects.filter(
                title='a question').exists())


class CourseEditViewTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.factory = APIRequestFactory()
        self.view = views.CourseView.as_view()

        self.setup_database()

    def test_deleting_question(self):
```

```
        courseData = {'name': 'edit_1', [...],
                      'language': 'en'}

        course = serializers.CourseSerializer(data=courseData)
        if not course.is_valid():
            self.assertTrue(False)
        course.create(courseData)
        self.assertTrue(models.Course.objects.filter(name='edit_1').exists())

        request = self.factory.get('/courses/')
        request.user = self.u1
        edit = serializers.CourseEditSerializer(
            models.Course.objects.filter(name='edit_1').first()).data

        del edit['modules'][0]['questions'][1]

        import json
        data = json.loads(json.dumps(edit))

        data['modules'][0]['questions'][0]['answers'] = \
            data['modules'][0]['questions'][0]['question_body']['answers']

        del data['modules'][0]['questions'][0]['question_body']

        data['modules'][0]['questions'][0]['order'] = 0

        data['responsible_mod'] = self.u1
        course = serializers.CourseSerializer(data=data)
        if not course.is_valid():
            self.assertTrue(False)
        course.create(data)

        self.assertFalse(models.Question.objects.filter(
            title='this one will be removed').exists())

    def test_deleting_module(self):
        courseData = {'name': 'edit_2', [...],
                      'language': 'en'}

        course = serializers.CourseSerializer(data=courseData)
        if not course.is_valid():
            self.assertTrue(False)
        course.create(courseData)
        self.assertTrue(models.Course.objects.filter(name='edit_2').exists())

        request = self.factory.get('/courses/')
```

```
        request.user = self.u1
        edit = serializers.CourseEditSerializer(
            models.Course.objects.filter(name='edit_2').first()).data

        del edit['modules'][1]

        import json
        data = json.loads(json.dumps(edit))

        data['modules'][0]['questions'][0]['answers'] = \
            data['modules'][0]['questions'][0]['question_body']['answers']

        del data['modules'][0]['questions'][0]['question_body']

        data['modules'][0]['questions'][0]['order'] = 0

        data['responsible_mod'] = self.u1
        course = serializers.CourseSerializer(data=data)
        if not course.is_valid():
            self.assertTrue(False)
        course.create(data)

        self.assertFalse(
            models.Module.objects.filter(name='another module').exists())


class RequestViewTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.factory = APIRequestFactory()
        self.view = views.RequestView.as_view()

        self.mod_group = Group(name='moderator')
        self.mod_group.save()

        self.u1 = User(username='user1')
        self.u1.save()
        self.u1_profile = models.Profile(user=self.u1)
        self.u1_profile.save()
        self.u2 = User.objects.create(username='mod')
        self.u2.groups.add(self.mod_group)
        self.u2.save()
        self.u2_profile = models.Profile(user=self.u2)
        self.u2_profile.save()
        self.u3 = User.objects.create(username='spamer')
        self.u3.save()
        self.u3_profile = models.Profile(user=self.u3)
```

```
        self.u3_profile.save()
        self.u3.profile.last_modrequest = timezone.localdate()

        Group(name='admin').save()

    def test_get(self):
        # Test for true positive
        request_1 = self.factory.get('/user/can_request_mod')
        force_authenticate(request_1, self.u1)
        response = self.view(request_1)
        self.assertEqual(response.status_code, 200)
        self.assertEqual(response.data, {'allowed': True})

        # Test for true negative
        request_2 = self.factory.get('/user/can_request_mod')
        force_authenticate(request_2, self.u2)
        response = self.view(request_2)
        self.assertEqual(response.status_code, 200)
        self.assertFalse(not response.data)

        # Test for true negative
        request_3 = self.factory.get('/user/can_request_mod')
        force_authenticate(request_3, self.u3)
        response = self.view(request_3)
        self.assertEqual(response.status_code, 200)
        self.assertEqual(response.data, {'allowed': False})

    def test_post(self):
        request_1 = self.factory.post('user/request_mod',
                                      {'reason': 'you need me'}, format='json')
        force_authenticate(request_1, self.u1)
        response = self.view(request_1)
        self.assertEqual(response.status_code, 200)
        self.assertEqual(response.data, {'Request': 'ok'})
        self.assertFalse(self.u1.profile.modrequest_allowed())

        request_2 = self.factory.post('user/request_mod',
                                      {'reason': 'you need me'}, format='json')
        force_authenticate(request_2, self.u2)
        response = self.view(request_2)
        self.assertEqual(response.status_code, 403)
        self.assertTrue(self.mod_group in self.u2.groups.all())

        request_3 = self.factory.post('user/request_mod',
                                      {'reason': 'you need me'}, format='json')
        force_authenticate(request_3, self.u3)
```

```
        response = self.view(request_3)
        self.assertEqual(response.status_code, 403)
        self.assertFalse(self.mod_group in self.u1.groups.all())


class UserRightsViewTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.factory = APIRequestFactory()
        self.view = views.UserRightsView.as_view()

        self.mod_group = Group.objects.create(name='moderator')
        self.admin_group = Group.objects.create(name='admin')

        self.u1 = User.objects.create_user(username='user1')
        self.u1_profile = models.Profile.objects.create(user=self.u1)

        self.u2 = User.objects.create_user(username='mod')
        self.u2.groups.add(self.mod_group)
        self.u2.save()
        self.u2_profile = models.Profile.objects.create(user=self.u2)

        self.u3 = User.objects.create_user(username='admin')
        self.u3.groups.add(self.admin_group)
        self.u3.save()
        self.u3_profile = models.Profile.objects.create(user=self.u3)

        self.u4 = User.objects.create_user(username='spamer')
        self.u4_profile = models.Profile.objects.create(user=self.u4)
        self.u4.profile.last_modrequest = timezone.localdate()

        self.users = [self.u1, self.u2, self.u3, self.u4]
        # bad users are those who aren't allowed to promote users
        self.bad_users = [self.u1, self.u2, self.u4]

    def test_post(self):
        # check if 403 is correctly thrown
        requests = []
        responses = []
        i = 0
        for request_user in self.bad_users:
            requests.append(self.factory.post(
                'user/' + str(self.u1.id) + '/rights',
                {'right': 'admin', 'action': 'promote'},
                format='json')
            )
            force_authenticate(requests[i], request_user)
```

```
        responses.append(self.view(requests[i]))
        self.assertEqual(responses[i].status_code, 403)
        self.assertFalse(self.u1.profile.is_admin())
        i += 1

    # try withdrawing admin rights from every kind of user as an admin
    # it should always work and the user to demote should not be
    # in the admin group afterwards
    for user_to_change in self.users:
        # try withdrawing modrights
        request2 = (self.factory.post(
            'user/' + str(user_to_change.id) + '/rights/',
            {'right': 'moderator', 'action': 'demote'},
            format='json'
        ))
        force_authenticate(request2, self.u3)
        response2 = (self.view(request2, user_id=user_to_change.id))
        self.assertEqual(response2.status_code, 200)
        self.assertFalse(
            user_to_change.groups.filter(name='moderator').exists()
        )

        # try withdrawing admin rights
        request1 = (self.factory.post(
            'user/' + str(user_to_change.id) + '/rights/',
            {'right': 'admin', 'action': 'demote'},
            format='json'
        ))
        force_authenticate(request1, self.u3)
        response1 = (self.view(request1, user_id=user_to_change.id))
        self.assertEqual(response1.status_code, 200)
        self.assertFalse(
            user_to_change.groups.filter(name='admin').exists()
        )

        # return adminrights to the admin user if they were
        # successfully withdrawn
        if (not self.u3_profile.is_admin()):
            self.u3.groups.add(self.admin_group)

        # try granting modrights
        request3 = (self.factory.post(
            'user/' + str(user_to_change.id) + '/rights/',
            {'right': 'moderator', 'action': 'promote'},
            format='json'
        ))
```

```
        force_authenticate(request3, self.u3)
        response3 = (self.view(request3, user_id=user_to_change.id))
        self.assertEqual(response3.status_code, 200)
        self.assertTrue(
            user_to_change.groups.filter(name='moderator').exists()
        )

        # try granting admin rights
        request4 = (self.factory.post(
            'user/' + str(user_to_change.id) + '/rights/',
            {'right': 'admin', 'action': 'promote'},
            format='json'
        ))
        force_authenticate(request4, self.u3)
        response4 = (self.view(request4, user_id=user_to_change.id))
        self.assertEqual(response4.status_code, 200)
        self.assertTrue(
            user_to_change.groups.filter(name='admin').exists()
        )


class TryTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.factory = APIRequestFactory()
        self.view = views.QuestionView.as_view()
        self.setup_database()

        courseData = {'name': 'quiz_1', [...],
                      'language': 'en'}

        course = serializers.CourseSerializer(data=courseData)
        course.create(courseData)

    def test_check_try(self):
        # creation is possible and quiz query is sorted

        course = models.Course.objects.filter(name='quiz_1')
        self.assertTrue(course.exists())

        course = course.first()
        # create one true
        question = MultipleChoice.models.MultipleChoiceAnswer.objects.filter(
            question__module__course__name='quiz_1', is_correct=True).first()
        correct_answer = self.factory.post(
            'courses/' + str(course.id) + '/0/0/',
            {'answers': [question.id]}, format='json')
```

```
        force_authenticate(correct_answer, self.u1)

        response = views.QuestionView.as_view()(correct_answer,
                                                course_id=course.id,
                                                module_id=0, question_id=0)
        # create one false
        question = MultipleChoice.models.MultipleChoiceAnswer.objects.filter(
            question__module__course__name='quiz_1', is_correct=False).first()
        false_answer = self.factory.post(
            'courses/' + str(course.id) + '/0/0/',
            {'answers': [question.id]}, format='json')
        force_authenticate(false_answer, self.u1)

        response = views.QuestionView.as_view()(false_answer,
                                                course_id=course.id,
                                                module_id=0, question_id=0)

        get_statistics = self.factory.get('user/statistics/')
        force_authenticate(get_statistics, self.u1)

        response = views.StatisticsView.as_view()(get_statistics)

        self.assertEqual(len(response.data), 2)

    def test_date(self):
        # initalize the database with 2 trys

        self.test_check_try()
        # check date field
        from django.utils import timezone
        from datetime import timedelta

        now = timezone.now()
        end = (now + timedelta(days=+1)).strftime('%Y-%m-%d %H:%M:%S.%f')
        start = (now + timedelta(days=-1)).strftime('%Y-%m-%d %H:%M:%S.%f')

        get_statistics = self.factory.post('user/statistics/',
                                            {'id': self.u1.id,
                                             'date': {'start': start,
                                                      'end': end}},
                                            format='json')
        force_authenticate(get_statistics, self.u1)

        response = views.StatisticsView.as_view()(get_statistics)
        self.assertEqual(len(response.data), 2)
```

```
        # check for to old dates
        end = (now + timedelta(days=-2)).strftime('%Y-%m-%d %H:%M:%S.%f')
        start = (now + timedelta(days=-7)).strftime('%Y-%m-%d %H:%M:%S.%f')

        get_statistics = self.factory.post('user/statistics/',
                                    {'id': self.u1.id,
                                     'date': {'start': start,
                                              'end': end}},
                                    format='json')
        force_authenticate(get_statistics, self.u1)

        response = views.StatisticsView.as_view()(get_statistics)
        self.assertEqual(len(response.data), 0)


class QuizTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.factory = APIRequestFactory()
        self.view = views.QuestionView.as_view()
        self.setup_database()

        courseData = {'name': 'quiz_1', [...],
                      'language': 'en'}

        course = serializers.CourseSerializer(data=courseData)
        if not course.is_valid():
            self.assertTrue(False)
        course.create(courseData)

    def test_create_quiz(self):
        # creation is possible and quiz query is sorted

        course = models.Course.objects.filter(name='quiz_1')
        self.assertTrue(course.exists())
        course = course.first()
        self.assertEqual(len(course.quizquestion_set.all()), 5)
        self.assertEqual(course.quizquestion_set.all()[0].question, 'first')
        self.assertEqual(course.quizquestion_set.all()[4].question, 'last')

        # try accesing the quiz before answering the questions is not valid

        request = self.factory.get('courses/' + str(course.id) + '/quiz/',
                                   format='json')
        force_authenticate(request, self.u1)

        response = views.QuizView.as_view()(request, course_id=course.id)
```

```
        self.assertEqual(response.status_code, 403)

        # check if return value after every question in course is done is
        # correct
        question = MultipleChoice.models.MultipleChoiceAnswer.objects.filter(
            question__module__course__name='quiz_1', is_correct=True).first()
        correct_answer = self.factory.post(
            'courses/' + str(course.id) + '/0/0/',
            {'answers': [question.id]}, format='json')
        force_authenticate(correct_answer, self.u1)

        response = views.QuestionView.as_view()(correct_answer,
                                                course_id=course.id,
                                                module_id=0, question_id=0)
        self.assertEqual(response.data['next'], 'quiz')

        # send post with false or correct answer will return 200 and send to
        # next quiz question

        answer_correct = []
        i = 0
        for ans in course.quizquestion_set.all():

            answers = []
            for quiz in ans.quizanswer_set.all():
                if i is 1:
                    answers.append({'chosen': not quiz.correct, 'id': quiz.id})
                else:
                    answers.append({'chosen': quiz.correct, 'id': quiz.id})

            answer_correct.append({"answers": answers, 'id': ans.id})
            i += 1

        correct_answer = self.factory.post(
            'courses/' + str(course.id) + '/quiz/',
            {"type": "check_answers", "answers": answer_correct},
            format='json')
        force_authenticate(correct_answer, self.u1)

        response = views.QuizView.as_view()(correct_answer,
                                            course_id=course.id,
                                            )

        # return value of correct answer
        self.assertEqual(response.status_code, 200)
```

```
        self.assertTrue(response.data[0]['solved'])
        self.assertFalse(response.data[1]['solved'])

        answer_wrong = models.QuizAnswer.objects.filter(
            correct=False,
            quiz=course.quizquestion_set.all()[
                0])
        wrong_answer = self.factory.post(
            'courses/' + str(course.id) + '/quiz/0/',
            {'answers': [(lambda x: x.id)(x) for x in answer_wrong]},
            format='json')
        force_authenticate(wrong_answer, self.u1)

        response = views.QuestionView.as_view()(wrong_answer,
                                                course_id=course.id,
                                                module_id=0, question_id=0)

        # return value of wrong answer
        self.assertEqual(response.status_code, 200)

        # creation for unsolvable quiz resolves in error
        courseData = {'name': 'quiz_2', [...],
                      'language': 'en'}

        quiz = serializers.CourseSerializer(data=courseData)

        self.assertTrue(quiz.is_valid())
        with self.assertRaises(ParseError):
            quiz.create(courseData)
        self.assertFalse(models.Course.objects.filter(name='quiz_2').exists())


class ProfileTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.setup_database()

    def test_unique_hash(self):
        hash_u1_1 = self.u1_profile.get_hash()
        u2 = User(username='u2')
        u2.save()
        u2_profile = Profile(user=u2)
        u2_profile.save()
        hash_u1_2 = self.u1_profile.get_hash()
        hash_u2_1 = u2_profile.get_hash()
```

```
        self.assertTrue(hash_u1_1 == hash_u1_2)
        self.assertFalse(hash_u2_1 == hash_u1_1)

    def test_changed_profile_hash(self):
        hash_u1_1 = self.u1_profile.get_hash()
        self.u1_profile.ranking = 100
        self.u1_profile.save()
        hash_u1_2 = self.u1_profile.get_hash()

        self.assertTrue(hash_u1_1 == hash_u1_2)


class QuestionViewTest(DatabaseMixin, TestCase):
    def setUp(self):
        self.factory = APIRequestFactory()
        self.view = views.QuestionView.as_view()

        self.setup_database()

    def test_get(self):
        # Test for true positive
        request_1 = self.factory.get('/course/1/1/1')
        force_authenticate(request_1, self.u1)
        response = self.view(request_1, course_id=1, module_id=0,
                             question_id=0)
        self.assertEqual(response.status_code, 200)
        self.assertEqual(response.data,
                         serializers.QuestionSerializer(self.q1_test, context={
                             'request': request_1}).data)

        # Test for true negative
        response = self.view(request_1, course_id=1, module_id=0,
                             question_id=128)
        self.assertEqual(response.status_code, 404)

        # Test for outer catch
        response = self.view(request_1, course_id=128, module_id=128,
                             question_id=128)
        self.assertEqual(response.status_code, 404)

        # Test for can't access
        response = self.view(request_1, course_id=1, module_id=0,
                             question_id=1)
        self.assertEqual(response.status_code, 403)

    def test_post(self):
```

```
        request_1 = self.factory.post('', {'answers': [0, 1]})
        request_1.user = self.u1
        force_authenticate(request_1, self.u1)
        response = self.view(request_1, course_id=1, module_id=0,
                             question_id=0)

        self.assertEqual(response.status_code, 200)
        self.assertEqual(response.data, {'evaluate': False})

        can = views.QuestionView().can_access_question(self.u1, self.q2_test,
                                                       1, 1)
        self.assertFalse(can)

        # Test doesn't work because of weird behavior of testing API
        #
        # request_1 = self.factory.post('', {'answers': [2]})
        # request_1.user = self.u1
        # force_authenticate(request_1, self.u1)
        # response = self.view(request_1, course_id=1, module_id=0,
        # question_id=0)

        # can = views.QuestionView().can_access_question(self.u1, self.q2_test)
        # self.assertTrue(can)

    def test_can_access_question(self):
        can = views.QuestionView().can_access_question(self.u1, self.q1_test,
                                                       0, 0)
        self.assertTrue(can)

        can = views.QuestionView().can_access_question(self.u1, self.q2_test,
                                                       2, 1)
        self.assertFalse(can)
```

F.2 Protractor/Selenium Testklasse

```
// spec.js
describe('Login Verification', function() {
  beforeEach(function() {
    browser.get('http://localhost:4200');
  });

  it('should only login with verified credentials', function() {
    expect(browser.getTitle()).toEqual('Clonecademy');
    browser.waitForAngular();
    let elem = element.all(by.css('input'));
```

```
    expect(elem.count()).toEqual(2);
    elem.get(0).sendKeys("iliricon");
    elem.get(1).sendKeys("Apfelbaum");
    elem.get(1).sendKeys(protractor.Key.ENTER);
    expect(element(by.css(".error-msg")).isDisplayed()).toBeTruthy();
  });
  it('it should login with verified credentials', function() {
    expect(browser.getTitle()).toEqual('Clonecademy');
    browser.waitForAngular();
    let elem = element.all(by.css('input'));
    expect(elem.count()).toEqual(2);
    elem.get(0).sendKeys("admin");
    elem.get(1).sendKeys("Apfelbaum");
    elem.get(1).sendKeys(protractor.Key.ENTER);
    browser.waitForAngular();
    expect(element(by.css(".sidebar")).isDisplayed()).toBeTruthy();
    elem = element.all(by.css('.mat-tab-link'));
    elem.get(2).click().then(function() {
      var EC = protractor.ExpectedConditions;
      elem = element.all(by.css('button'));
      expect(elem.count()).toEqual(2);
      browser.wait(EC.elementToBeClickable(elem), 30000).then(function () {
                 elem.click();
      });
      let div = element(by.css('cdk-overlay-15'));
      elem = element.all(by.css('button')).get(0);
      browser.sleep(3000);
      browser.waitForAngular();
      elem.click();
      browser.waitForAngular();
    });
  });

});

describe('Admin Tests', function() {
  beforeAll(function() {
    browser.get('http://localhost:4200');
    expect(browser.getTitle()).toEqual('Clonecademy');
    browser.waitForAngular();
    let elem = element.all(by.css('input'));
    elem.get(0).sendKeys("admin");
    elem.get(1).sendKeys("Apfelbaum");
    elem.get(1).sendKeys(protractor.Key.ENTER);
    browser.waitForAngular();
    expect(element(by.css(".sidebar")).isDisplayed()).toBeTruthy();
```

```
  });
  it ('should create a new course', function() {
    element(by.css('.addCourse')).click()
  });
});
```

F.3 Coverage

Hier angefügt ist ein vollständiger Coverage-Report.

Coverage report: 90%

filter...

Module ↓	*statements*	*missing*	*excluded*	*coverage*
django/clonecademy/__init__.py	0	0	0	100%
django/clonecademy/settings.py	25	0	0	100%
django/clonecademy/urls.py	7	0	0	100%
django/learning_base/__init__.py	0	0	0	100%
django/learning_base/admin.py	16	0	0	100%
django/learning_base/custom_permissions.py	14	1	0	93%
django/learning_base/default_picture.py	1	0	0	100%
django/learning_base/info/__init__.py	0	0	0	100%
django/learning_base/info/models.py	41	3	0	93%
django/learning_base/info/serializer.py	22	5	0	77%
django/learning_base/models.py	149	8	0	95%
django/learning_base/multiple_choice/__init__.py	0	0	0	100%
django/learning_base/multiple_choice/models.py	38	3	0	92%
django/learning_base/multiple_choice/serializer.py	51	1	0	98%
django/learning_base/serializers.py	299	32	0	89%
django/learning_base/tests.py	631	5	0	99%
django/learning_base/views.py	456	117	0	74%
django/manage.py	13	6	0	54%
Total	**1763**	**181**	**0**	**90%**

coverage.py v4.4.1, created at 2017-09-28 15:34

Coverage for **django/learning_base/info/models.py** : 93%

41 statements | 38 run | 3 missing | 0 excluded

```
1  """
2  Models for information type questions
3  """
4  from django.db import models
5  from learning_base.models import Question
6
7
8  class InformationText(Question):
9      """
10     A 'question' that only displays an informative text
11     to the user. It requires no input and is only used to
12     convey additional learning material to the reader.
13     :author: Claas Voelcker
14     """
15
16     __name__ = "info_text"
17
18     image = models.TextField(
19         blank=True,
20     )
21
22     text_field = models.TextField()
23
24     @staticmethod
25     def not_solvable():
26         """
27         always solvable
28         :author: Claas Voelcker
29         :return: False
30         """
31         return False
32
33     @staticmethod
34     def evaluate(data):
35         """
36         always correctly solved
37         :author: Claas Voelcker
38         :return: True
39         """
40         return True
41
42     @staticmethod
43     def num_correct_answers():
44         """
45         has no answers
46         :author: Claas Voelcker
47         """
48         return 0
49
50     @staticmethod
51     def get_serializer():
52         """
53         returns the correct serializer
54         :author: Claas Voelcker
55         :return: serializer for the question
56         """
57         from . import serializer
58         return serializer.InformationTextSerializer
59
```

```
60      @staticmethod
61      def get_edit_serializer():
62          """
63          returns the serializer for editing
64          :author: Claas Voelcker
65          :return: serializer for the question
66          """
67          return InformationText.get_serializer()
68
69      @staticmethod
70      def get_points():
71          """
72          has no points for a correct answer
73          :author: Claas Voelcker
74          """
75          return 0
76
77      def __str__(self):
78          return "Learning text {}".format(self.id)
79
80
81  class InformationYoutube(Question):
82      """
83      A 'question' that only displays an informative text
84      to the user. It requires no input and is only used to
85      convey additional learning material to the reader.
86      @author: Claas Voelcker
87      """
88
89      __name__ = "info_text_youtube"
90
91      url = models.TextField(
92          blank=True,
93      )
94
95      text_field = models.TextField()
96
97      @staticmethod
98      def not_solvable():
99          """
100         always solvable
101         :author: Claas Voelcker
102         :return: False
103         """
104         return False
105
106     @staticmethod
107     def evaluate(data):
108         """
109         always correctly solved
110         :author: Claas Voelcker
111         :return: True
112         """
113
114         return True
115
116     @staticmethod
117     def num_correct_answers():
118         """
119         has no answers
120         :author: Claas Voelcker
121         """
122         return 0
123
124     @staticmethod
```

```
125     def get_serializer():
126         """
127         returns the correct serializer
128         :author: Claas Voelcker
129         :return: serializer for the question
130         """
131         from . import serializer
132         return serializer.InformationYoutubeSerializer
133 
134     @staticmethod
135     def get_edit_serializer():
136         """
137         returns the serializer for editing
138         :return: serializer for the question
139         """
140         from . import serializer
141         return serializer.InformationYoutubeSerializer
142 
143     @staticmethod
144     def get_points():
145         """
146         has no points for a correct answer
147         :author: Claas Voelcker
148         """
149         return 0
150 
151     def __str__(self):
152         """
153         :return: string representation from the id
154         """
155         return "Learning text {}".format(self.id)
```

« index coverage.py v4.4.1, created at 2017-09-28 15:34

Coverage for **django/learning_base/info/serializer.py** : 77%

22 statements | 17 run | 5 missing | 0 excluded

```
1  """
2  Serializers for the information type questions
3  """
4  from re import compile
5
6  from rest_framework import serializers
7  from .models import InformationYoutube, InformationText
8
9
10 class InformationTextSerializer(serializers.ModelSerializer):
11     """
12     The serializer for information text type questions.
13     @author: Claas Voelcker
14     """
15
16     class Meta:
17         model = InformationText
18         fields = ('text_field', 'image')
19
20     def create(self, validated_data):
21         """
22         creates a InformationText from validated data
23         :param validated_data: the data for the new db entry
24         """
25         question = InformationText(**validated_data)
26         question.module = validated_data['module']
27         question.save()
28
29
30 class InformationYoutubeSerializer(serializers.ModelSerializer):
31     """
32     The serializer for information video type questions.
33     @author: Claas Voelcker
34     """
35
36     # ID extraction pattern
37     pattern = compile(
38         r'(?:http(?:s)?)?:\/\/(?:www\.)?(?:youtu\.be|youtube\.com)?\/(?:watch\?v=|embed\/)?(.*)'
39
40     class Meta:
41         model = InformationYoutube
42         fields = ('text_field', 'url')
43
44     def create(self, validated_data):
45         """
46         Creates the object. Takes a valid youtube url and extracts the video
47         id. This makes it possible to
48         :param validated_data: the JSON containing all data
49         :return: True for a valid serialization
50         """
51         question = InformationYoutube(**validated_data)
52         question.url = self.pattern.findall(validated_data['url'])[0]
53         question.module = validated_data['module']
54         question.save()
55         return True
```

« index coverage.py v4.4.1, created at 2017-09-28 15:34

Coverage for **django/learning_base/models.py** : 95%

149 statements | 141 run | 8 missing | 0 excluded

```
1  """
2  This module contains all database models not provided by django
3  itself.
4  :author: Claas Voelcker
5  """
6
7  from hashlib import sha512
8  from django.db import models
9  from django.contrib.auth.models import User
10 from django.utils import timezone
11 from polymorphic.models import PolymorphicModel
12
13 from .default_picture import default_picture
14
15
16 class Profile(models.Model):
17     """
18     A user profile that stores additional information about a user
19     :author: Claas Voelcker
20     """
21
22     class Meta:
23         ordering = ('ranking',)
24
25     user = models.OneToOneField(
26         User,
27         on_delete=models.CASCADE,
28     )
29
30     birth_date = models.DateField(
31         blank=True,
32         null=True,
33     )
34
35     last_modrequest = models.DateField(
36         blank=True,
37         null=True,
38     )
39
40     language = models.CharField(
41         verbose_name='Language',
42         max_length=2,
43         default="en"
44     )
45
46     avatar = models.TextField(
47         verbose_name="Avatar of the User",
48         default=default_picture,
49         null=True,
50         blank=True,
51     )
52
53     ranking = models.IntegerField(
54         default=0
55     )
56
57     def get_link_to_profile(self):
58         """
59         :return: the link to the users profile page
```

```
60          """
61          return "clonecademy.net/admin/profiles/{}/".format(self.user.id)
62
63      def modrequest_allowed(self):
64          """
65          :return: True if the user is allowed to request moderator rights
66          """
67          return (not self.is_mod()
68                  and (self.last_modrequest is None
69                       or (timezone.now() - self.last_modrequest).days >= 7))
70
71      def is_mod(self):
72          """
73          :return: True if the user is in the group moderators
74          """
75          return self.user.groups.filter(name="moderator").exists()
76
77      def is_admin(self):
78          """
79          Returns True if the user is in the group admin
80          :return: whether the user belong to the admin group
81          """
82          return self.user.groups.filter(name="admin").exists()
83
84      def get_hash(self):
85          """
86          calculates a hash to get anonymous user data
87          :return: the first 10 digits of the hash
88          """
89          return sha512(str.encode(self.user.username)).hexdigest()[:10]
90
91      def __str__(self):
92          return str(self.user)
93
94
95  class CourseCategory(models.Model):
96      """
97      The type of a course, meaning the field in which the course belongs, e.g.
98      biochemistry, cloning, technical details.
99      """
100     name = models.CharField(
101         help_text="Name of the category (e.g. biochemistry)",
102         max_length=144,
103         unique=True,
104     )
105
106     color = models.CharField(
107         help_text="Color that is used in the category context",
108         max_length=7,
109         default="#000000"
110     )
111
112     def __str__(self):
113         return self.name
114
115
116 class Course(models.Model):
117     """
118     One course is a group of questions which build on each other and should be
119     solved together. These questions should have similar topics, difficulty
120     and should form a compete unit for learning.
121     :author: Claas Voelcker
122     """
123
124     class Meta:
```

```
125         unique_together = ['category', 'name']
126
127     # difficulty selection and mapping to human readable names
128     EASY = 0
129     MODERATE = 1
130     DIFFICULT = 2
131     EXPERT = 3
132     DIFFICULTY = (
133         (EASY, 'Easy (high school students)'),
134         (MODERATE, 'Moderate (college entry)'),
135         (DIFFICULT, 'Difficult (college students'),
136         (EXPERT, 'Expert (college graduates)')
137     )
138
139     # language selection and mapping
140     GER = 'de'
141     ENG = 'en'
142     LANGUAGES = (
143         (GER, 'German/Deutsch'),
144         (ENG, 'English')
145     )
146
147     # the name of the course
148     name = models.CharField(
149         verbose_name='Course name',
150         help_text="A short concise name for the course",
151         max_length=144
152     )
153
154     # foreign key mapping to the CourseCategory object
155     category = models.ForeignKey(
156         CourseCategory,
157         null=True,
158         blank=True
159     )
160
161     # choice field mapped to dictionary above
162     difficulty = models.IntegerField(
163         verbose_name='Course difficulty',
164         choices=DIFFICULTY,
165         default=MODERATE
166     )
167
168     # choice field mapped to dictionary above
169     language = models.CharField(
170         verbose_name='Course Language',
171         max_length=2,
172         choices=LANGUAGES,
173         default=ENG
174     )
175
176     # foreign key mapping to the user who can edit the course
177     responsible_mod = models.ForeignKey(
178         User,
179         on_delete=models.SET_NULL,
180         null=True,
181         blank=True
182     )
183
184     # should the course be serialized for normal users
185     is_visible = models.BooleanField(
186         verbose_name='Is the course visible',
187         default=False
188     )
189
```

```
190     # a short description of the course
191     description = models.CharField(
192         max_length=144,
193         null=True,
194         blank=True,
195         default=""
196     )
197 
198     def __str__(self):
199         return self.name
200 
201     def num_of_modules(self):
202         """
203         Returns the number of modules
204         """
205         return len(Module.objects.filter(course=self))
206 
207 
208 class Module(models.Model):
209     """
210     A Course is made out of several modules and a module contains the questions
211     """
212 
213     class Meta:
214         unique_together = ['order', 'course']
215         ordering = ['order']
216 
217     name = models.CharField(
218         help_text="A short concise name for the module",
219         verbose_name='Module name',
220         max_length=144
221     )
222 
223     learning_text = models.TextField(
224         help_text="The learning Text for the module",
225         verbose_name="Learning text"
226     )
227 
228     course = models.ForeignKey(
229         Course,
230         on_delete=models.CASCADE
231     )
232 
233     order = models.IntegerField()
234 
235     description = models.CharField(
236         max_length=144,
237         null=True,
238         blank=True
239     )
240 
241     def __str__(self):
242         return self.name
243 
244     def num_of_questions(self):
245         """
246         Returns the number of questions in the module
247         """
248         return len(self.question_set.all())
249 
250     def get_previous_in_order(self):
251         """
252         Gets the previous module in the ordering
253         :return: the previous module in the same course
254         """
```

```
255         modules = self.course.module_set.all()
256         if list(modules).index(self) <= 0:
257             return False
258         return modules[list(modules).index(self) - 1]
259 
260     def is_first_module(self):
261         """
262         checks whether the given module is the first in a course
263         :return: True, iff this module has the lowest order in the course
264         """
265         modules = self.course.module_set
266         return self == modules.first()
267 
268     def is_last_module(self):
269         """
270         Returns True if this is the final module in a course
271         """
272         modules = self.course.module_set
273         return self == modules.last()
274 
275 
276 class Question(PolymorphicModel):
277     """
278     A question is the smallest unit of the learning process. A question has a
279     task that can be solved by a user, a correct solution to evaluate the
280     answer and a way to provide feedback to the user.
281 
282     :author: Claas Voelcker
283     """
284 
285     class Meta:
286         unique_together = ['module', 'order']
287         ordering = ['module', 'order']
288 
289     # a title for the question
290     title = models.TextField(
291         verbose_name='Question title',
292         help_text="A short and concise name for the question",
293         blank=True,
294         null=True
295     )
296 
297     # the question text, that provides additional information to the user
298     text = models.TextField(
299         verbose_name='Question text',
300         help_text="This field can contain markdown syntax"
301     )
302 
303     # the specific question
304     question = models.TextField(
305         verbose_name='Question',
306         help_text="This field can contain markdown syntax",
307         blank=True,
308         null=True
309     )
310 
311     # a custom feedback that can be displayed
312     feedback = models.TextField(
313         verbose_name="feedback",
314         help_text="The feedback for the user after a sucessful answer",
315         blank=True,
316         null=True
317     )
318 
319     # the ordering attribute of the question (needs to be explicitly saved)
```

```
320     order = models.IntegerField()
321
322     # foreign key mapping to the module that contains this question
323     module = models.ForeignKey(
324         Module,
325         verbose_name="Module",
326         help_text="The corresponding module for the question",
327         on_delete=models.CASCADE
328     )
329
330     def is_first_question(self):
331         """
332         Checks whether this is the first question in the module
333
334         :author: Claas Voelcker
335         :return: whether this is the first question or not
336         """
337         questions = self.module.question_set
338         return self == questions.first()
339
340     def is_last_question(self):
341         """
342         Checks whether this is the last question in the module
343
344         :author: Claas Voelcker
345         :return: whether this is the last question or not
346         """
347         questions = self.module.question_set
348         return self == questions.last()
349
350     def get_previous_in_order(self):
351         """
352         Returns the previous question in the course
353         :author: Claas Voelcker
354         :return: the previous question in the same module
355         """
356         questions = self.module.question_set.all()
357         if list(questions).index(self) <= 0:
358             return False
359         return questions[list(questions).index(self) - 1]
360
361     def get_points(self):
362         """
363         Returns the number of ranking points for the question.
364         This method needs to be overridden by subclasses and
365         remains unimplemented here.
366         :author: Claas Voelcker
367         :return: the points
368         :raise: not implemented error
369         """
370         raise NotImplementedError
371
372     def __str__(self):
373         return self.title
374
375
376 class QuizQuestion(models.Model):
377     """
378     single Quiz Question with possible multiple answers
379     @author Leonhard Wiedmann
380     """
381     question = models.TextField(
382         verbose_name="quizQuestion",
383         help_text="The Question of this quiz question.",
384         default=""
```

```
385     )
386
387     image = models.TextField(
388         help_text="The image which is shown in this quiz",
389         default="",
390         blank=True
391     )
392
393     course = models.ForeignKey(
394         Course,
395         help_text="The Course of this question",
396         on_delete=models.CASCADE
397     )
398
399     def evaluate(self, data):
400         """
401         Checks whether the quiz question is answered correctly
402         :return: True iff all and only the correct answers are
403                  provided
404         """
405         answers = self.answer_set()
406         for ans in answers:
407             if ans.correct:
408                 for i in data['answers']:
409                     if 'id' in i and (i['id'] == ans.id and not i['chosen']):
410                         return False
411             if not ans.correct:
412                 for i in data:
413                     if 'id' in i and (i['id'] == ans.id and i['chosen']):
414                         return False
415         return True
416
417     def answer_set(self):
418         """
419         shortcut for all answers to a question
420         :return: all answers to the quizquestion
421         """
422         return self.quizanswer_set.all()
423
424     def is_solvable(self):
425         """
426         x
427         :return:
428         """
429         for ans in self.answer_set():
430             if ans.correct:
431                 return True
432         return False
433
434     def get_points(self):
435         """
436         returns the points for answering this question type
437         :return: 0 points
438         """
439         return 0
440
441
442 class QuizAnswer(models.Model):
443     """
444     Quiz answer with image and the value for correct answer
445     @author Leonhard Wiedmann
446     """
447     text = models.TextField(
448         help_text="The answer text"
449     )
```

```
450
451     img = models.TextField(
452         help_text="The image for this answer",
453         default="",
454         blank=True
455     )
456
457     correct = models.BooleanField(
458         help_text="If this answer is correct",
459         default=False
460     )
461
462     quiz = models.ForeignKey(QuizQuestion, on_delete=models.CASCADE)
463
464
465 class LearningGroup(models.Model):
466     """
467     A user group (currently not used)
468     """
469     name = models.CharField(
470         help_text="The name of the user group",
471         max_length=144)
472
473     def __str__(self):
474         return self.name
475
476
477 class Try(models.Model):
478     """
479     A try represents a submission of an answer. Each time an answer is
480     submitted, a Try object is created in the database, detailing answer,
481     whether it was answered correctly and the time of the submission.
482     :author: Claas Voelcker
483     """
484     user = models.ForeignKey(
485         User,
486         on_delete=models.SET_NULL,
487         null=True,
488     )
489
490     question = models.ForeignKey(
491         Question,
492         null=True,
493         on_delete=models.SET_NULL,
494     )
495
496     quiz_question = models.ForeignKey(
497         QuizQuestion,
498         null=True,
499         on_delete=models.SET_NULL,
500     )
501
502     answer = models.TextField(
503         verbose_name="The given answer",
504         help_text="The answers as pure string",
505         null=True
506     )
507
508     date = models.DateTimeField(
509         default=timezone.now,
510         null=True
511     )
512
513     solved = models.BooleanField(
514         default=False
```

```
515     )
516
517     def __str__(self):
518         return "Solution_{}_{}_{}".format(
519             self.question, self.solved, self.date)
520
521
522 def started_courses(user):
523     """
524     returns all courses started by a user
525     :param user: the user that is currently accessing the database
526     :return: all courses where the user has answered at least one course
527     """
528     courses = Course.objects.filter(
529         module__question__try__user=user)
530     return courses.distinct()
```

« index coverage.py v4.4.1, created at 2017-09-28 15:34

Coverage for
django/learning_base/multiple_choice/models.py : 92%

38 statements | 35 run | 3 missing | 0 excluded

```
1  """
2  module containing all models for multiple choice questions
3  """
4
5  from django.db import models
6  from learning_base.models import Question
7
8
9  class MultipleChoiceQuestion(Question):
10     """
11     A simple multiple choice question
12     :Leonhard Wiedmann:
13     """
14     __name__ = "multiple_choice"
15
16     question_image = models.TextField(
17         verbose_name="The Image for the question",
18         blank=True,
19     )
20
21     feedback_image = models.TextField(
22         verbose_name="The Image for the question",
23         blank=True,
24     )
25
26     def num_correct_answers(self):
27         """
28         :author: Claas Voelcker
29         :return: the number of correct answers
30         """
31         return len(MultipleChoiceAnswer.objects.filter(
32             question=self, is_correct=True))
33
34     def not_solvable(self):
35         """
36         Checks whether the question is actually solvable
37         :author: Claas Voelcker
38         :return: True if the number of answers is 0
39         """
40         return self.num_correct_answers() == 0
41
42     def evaluate(self, data):
43         """
44         data: Map of answers
45         :return: True if and only if the array of provided answers are exactly
46         the correct answers
47         :author: Tobias Huber
48         """
49
50         # get all correct answers, map them to their id and make a
51         # (mathematical) set out of it"
52         answers = set([x.id for x in
53                        self.multiplechoiceanswer_set.filter(is_correct=True)])
54         return answers == set(data)
55
56     def __str__(self):
57         return self.title
```

```
58
59      def answer_set(self):
60          """
61          shorthand for the answer set
62          :author: Leonhard Wiedmann
63          :return: the QuerySet of all answers
64          """
65          return self.multiplechoiceanswer_set.all()
66
67      @staticmethod
68      def get_serializer():
69          """
70          reverse for the serializer
71          :author: Claas Voelcker
72          :return: the fitting serializer
73          """
74          from . import serializer
75          return serializer.MultipleChoiceQuestionSerializer
76
77      def get_points(self):
78          """
79          returns the points value of the question
80          :return: 2 if the course is hard, else 1
81          """
82          return 2 if self.module.course.category == 2 else 1
83
84      @staticmethod
85      def get_edit_serializer():
86          """
87          reverse for the serializer
88          :author: Claas Voelcker
89          :return: the fitting serializer
90          """
91          from . import serializer
92          return serializer.MultipleChoiceQuestionEditSerializer
93
94
95  class MultipleChoiceAnswer(models.Model):
96      """
97      A possible answer to a multiple choice question
98      :author: Claas Voelcker
99      """
100     question = models.ForeignKey(
101         MultipleChoiceQuestion,
102         on_delete=models.CASCADE
103     )
104
105     text = models.TextField(
106         verbose_name="Answer text",
107         help_text="The answers text"
108     )
109
110     is_correct = models.BooleanField(
111         verbose_name='is the answer correct?',
112         default=False
113     )
114
115     img = models.TextField(
116         verbose_name="The Image for the answer",
117         blank=True
118     )
119
120     def __str__(self):
121         return self.text
122
```

```
123     @staticmethod
124     def get_serializer():
125         """
126         reverse for the serializer
127         :author: Claas Voelcker
128         :return: the fitting serializer
129         """
130         from learning_base.multiple_choice import serializer
131         return serializer.MultipleChoiceAnswerSerializer
132 
133     @staticmethod
134     def get_edit_serializer():
135         """
136         reverse for the serializer
137         :author: Claas Voelcker
138         :return: the fitting serializer
139         """
140         from learning_base.multiple_choice import serializer
141         return serializer.MultipleChoiceAnswerEditSerializer
```

« index coverage.py v4.4.1, created at 2017-09-28 15:34

Coverage for **django/learning_base/multiple_choice/serializer.py** : 98%

51 statements | 50 run | 1 missing | 0 excluded

```
1  """
2  serializers for MultipleChoice question types
3  """
4
5  from rest_framework import serializers
6  from rest_framework.exceptions import ParseError
7  from .models import MultipleChoiceAnswer, MultipleChoiceQuestion
8
9
10 class MultipleChoiceAnswerSerializer(serializers.ModelSerializer):
11     """
12     Serializer for MultipleChoice answers
13     :author: Leonhard Wiedmann
14     """
15     class Meta:
16         model = MultipleChoiceAnswer
17         fields = ('text', 'id', 'img')
18
19     def create(self, validated_data):
20         """
21         creation method for new db entries
22         :author: Leonhard Wiedmann
23         :param: validated_data valid data for the creation
24         """
25         answer = MultipleChoiceAnswer(**validated_data)
26         answer.question = validated_data['question']
27         answer.save()
28
29
30 class MultipleChoiceQuestionPreviewSerializer(serializers.ModelSerializer):
31     """
32     Serializer for MultipleChoice question preview
33     :author: Leonhard Wiedmann
34     """
35     class Meta:
36         model = MultipleChoiceQuestion
37         fields = ('body', "id",)
38
39
40 class MultipleChoiceAnswerEditSerializer(serializers.ModelSerializer):
41     """
42     Serializer for MultipleChoice answer editing
43     :author: Leonhard Wiedmann
44     """
45     class Meta:
46         model = MultipleChoiceAnswer
47         fields = ("text", "id", "is_correct", "img")
48
49
50 class MultipleChoiceQuestionEditSerializer(serializers.ModelSerializer):
51     """
52     Serializer for MultipleChoice question editing
53     :author: Leon Wiedmann
54     """
55     class Meta:
56         model = MultipleChoiceQuestion
```

```
57          fields = ("id", 'question_image', 'feedback_image')
58
59      def to_representation(self, obj):
60          """
61          responsible for json serialization of objects
62          :author: Leonhard Wiedmann
63          :param obj: the object that should be serialized
64          :return: a json representation of the object
65          """
66          values = super(MultipleChoiceQuestionEditSerializer,
67                         self).to_representation(obj)
68          answers = obj.answer_set()
69          values['answers'] = MultipleChoiceAnswerEditSerializer(answers,
70                                                                 many=True).data
71          return values
72
73
74  class MultipleChoiceQuestionSerializer(serializers.ModelSerializer):
75      """
76      Serializer for MultipleChoice questions editing
77      :author: Leon Wiedmann
78      """
79      class Meta:
80          model = MultipleChoiceQuestion
81          fields = ('id', 'question_image')
82
83      def to_representation(self, obj):
84          """
85          responsible for json serialization of objects
86          :author: Leonhard Wiedmann
87          :param obj: the object that should be serialized
88          :return: a json representation of the object
89          """
90          values = super(MultipleChoiceQuestionSerializer,
91                         self).to_representation(obj)
92          answers = obj.answer_set()
93          values['answers'] = MultipleChoiceAnswerSerializer(answers,
94                                                             many=True).data
95          return values
96
97      def create(self, validated_data):
98          """
99          creating new database entries while editing
100         :author: Leonhard Wiedmann
101         :param validated_data: valid data
102         """
103         answers = validated_data.pop('answers')
104         question = MultipleChoiceQuestion(**validated_data)
105         question.module = validated_data['module']
106         question.save()
107
108         for answer in answers:
109             answer['question'] = question
110             answer_serializer = MultipleChoiceAnswerSerializer(data=answer)
111             if not answer_serializer.is_valid():
112                 raise ParseError(detail=answer_serializer.errors, code=None)
113             else:
114                 answer_serializer.create(answer)
115         if question.not_solvable():
116             question.delete()
117             raise ParseError(
118                 detail="Unsolvable question {}".format(question.title),
119                 code=None)
```

Coverage for **django/learning_base/views.py** : 74%

456 statements | 339 run | 117 missing | 0 excluded

```
1  """
2  This module contains all directly accessed API functions
3  Views are not documented extensively in the code but at
4  https://github.com/Iliricon/clonecademy
5  """
6  from django.http import HttpResponse
7  from django.core.mail import send_mail
8  from django.contrib.auth.models import User, Group
9  from django.utils import timezone
10 from django.utils.crypto import get_random_string
11
12 from rest_framework import status
13 from rest_framework import authentication, permissions
14 from rest_framework.views import APIView
15 from rest_framework.exceptions import ParseError, PermissionDenied
16 from rest_framework.response import Response
17
18 from . import custom_permissions
19 from . import serializers
20 from .models import Course, CourseCategory, Try, Profile, started_courses
21
22
23 class CategoryView(APIView):
24     """
25     Shows, creates, updates and deletes a category
26     :author: Claas Voelcker, Tobias Huber
27     """
28     authentication_classes = (authentication.TokenAuthentication,)
29     permission_classes = (custom_permissions.IsAdminOrReadOnly,)
30
31     def get(self, request, format=None):
32         """
33         Shows the categories
34         :author: Claas Voelcker
35         :return: a list of all categories
36         """
37         categories = CourseCategory.objects.all()
38         data = serializers.CourseCategorySerializer(categories, many=True).data
39         return Response(data,
40                         status=status.HTTP_200_OK)
41
42     def post(self, request, format=None):
43         """
44         everything else but displaying
45         :author: Tobias Huber
46         """
47         data = request.data
48         # check if instance shall be deleted
49         if 'delete' in data and data['delete'] == 'true':
50             if 'id' in data:
51                 instance = CourseCategory.objects.get(id=data['id'])
52                 instance.delete()
53                 return Response(status=status.HTTP_204_NO_CONTENT)
54             return Response({'ans': 'a category with the given id'
55                                     + ' does not exist'},
56                             status=status.HTTP_404_NOT_FOUND)
57
58         # check if an id is given, signaling to update the corresponding cat.
59         if 'id' in data:
```

```
60              category_id = data['id']
61              if CourseCategory.objects.filter(id=category_id).exists():
62                  category = CourseCategory.objects.get(id=category_id)
63                  serializer = serializers.CourseCategorySerializer(
64                      category, data=data, partial=True, )
65              else:
66                  return Response(
67                      {'ans': 'a category with the id ' + str(category_id)
68                              + ' does not exist'},
69                      status=status.HTTP_404_NOT_FOUND)
70          else:
71              # else just create a plain serializer
72              serializer = serializers.CourseCategorySerializer(data=data)
73          if serializer.is_valid():
74              serializer.save()
75              return Response(serializer.data,
76                              status=status.HTTP_200_OK)
77          return Response(serializer.errors,
78                          status=status.HTTP_400_BAD_REQUEST)
79
80
81  class MultiCourseView(APIView):
82      """
83      View to see all courses of a language. The post method provides a general
84      interface with three filter settings.
85      @author Claas Voelcker
86      """
87      authentication_classes = (authentication.TokenAuthentication,)
88      permission_classes = (permissions.IsAuthenticated,)
89
90      def get(self, request, format=None):
91          """
92          Not implemented
93          """
94          return Response({'ans': 'Method not allowed'},
95                          status=status.HTTP_405_METHOD_NOT_ALLOWED)
96
97      def post(self, request, format=None):
98          """
99          Returns a set of courses detailed by the query. It expects a request
100         with the keys 'language', 'category', 'type'. The returning JSON
101         corresponds to the values. All values can be empty strings, resulting
102         in all courses being returned.
103         """
104         try:
105             types = ['mod', 'started']
106             categories = [str(x) for x in CourseCategory.objects.all()]
107             languages = [x[0] for x in Course.LANGUAGES]
108             data = request.data
109             r_type = data['type']
110             r_category = data['category']
111             r_lan = data['language']
112
113             # checks whether the query only contains acceptable keys
114             if not ((r_type in types or not r_type)
115                     and (r_category in categories or not r_category)
116                     and (r_lan in languages or not r_lan)):
117                 return Response({'ans': 'Query not possible'},
118                                 status=status.HTTP_400_BAD_REQUEST)
119
120             courses = Course.objects.all()
121             courses = courses.filter(language=r_lan)
122
123             # filter invisible courses if neccessary
124             if not (request.user.profile.is_mod()
125                     or request.user.profile.is_admin()):
```

```
126                 courses = courses.filter(is_visible=True)
127
128             if r_category != '':
129                 category = CourseCategory.objects.filter(
130                     name=r_category).first()
131                 courses = courses.filter(category=category)
132             if r_type == 'mod':
133                 courses = courses.filter(responsible_mod=request.user)
134             elif r_type == 'started':
135                 courses = started_courses(request.user)
136             data = serializers.CourseSerializer(courses, many=True, context={
137                 'request': request}).data
138             return Response(data, status=status.HTTP_200_OK)
139         except Exception as errors:
140             return Response({'ans': 'Query not possible' + str(errors)},
141                             status=status.HTTP_400_BAD_REQUEST)
142
143
144 class CourseEditView(APIView):
145     """
146     contains all the code related to edit a courses
147     TODO: this is probably redundant code
148     @author Leonhard Wiedmann
149     """
150     authentication_classes = (authentication.TokenAuthentication,)
151     permission_classes = (custom_permissions.IsModOrAdmin,)
152
153     def get(self, request, course_id=None, format=None):
154         """
155         Returns all the information about a course with the answers and the
156         solutions
157         """
158         if not course_id:
159             return Response({'ans': 'Method not allowed'},
160                             status=status.HTTP_405_METHOD_NOT_ALLOWED)
161
162         try:
163             course = Course.objects.filter(id=course_id).first()
164             course_serializer = serializers.CourseEditSerializer(
165                 course,
166                 context={
167                     'request': request})
168             data = course_serializer.data
169             return Response(data)
170
171         except Exception as errors:
172             return Response({'ans': str(errors)},
173                             status=status.HTTP_404_NOT_FOUND)
174
175     def post(self, request, course_id=None, format=None):
176         """
177         Not implemented
178         """
179         return Response({'ans': 'Method not allowed'},
180                         status=status.HTTP_405_METHOD_NOT_ALLOWED)
181
182
183 class CourseView(APIView):
184     """
185     Contains all code related to viewing and saving courses.
186     :author: Claas Voelcker
187     """
188     authentication_classes = (authentication.TokenAuthentication,)
189     permission_classes = (
190         custom_permissions.IsModOrAdminOrReadOnly,)
191
```

```
192     def get(self, request, course_id=None, format=None):
193         """
194         Returns a course if the course_id exists. The course, it's
195         modules and questions are serialized.
196 
197         :author: Claas Voelcker
198         :param request: request object containing auth token and user id
199         :param course_id: the id of the required course
200         :param format: unused (inherited)
201         :return: a response containing the course serialization
202         """
203 
204         # if no course id is given, the method was called wrong
205         if not course_id:
206             return Response({'ans': 'Method not allowed'},
207                             status=status.HTTP_405_METHOD_NOT_ALLOWED)
208         try:
209             # fetch the course object, serialize it and return
210             # the serialization
211             course = Course.objects.get(id=course_id)
212             course_serializer = serializers.CourseSerializer(course, context={
213                 'request': request})
214             return Response(course_serializer.data,
215                             status=status.HTTP_200_OK)
216         # in case of an exception, throw a "Course not found" error for the
217         # frontend, packaged in a valid response with an error status code
218         except Exception:
219             return Response({'ans': 'Course not found'},
220                             status=status.HTTP_404_NOT_FOUND)
221 
222     def post(self, request, course_id=None, format=None):
223         """
224         Saves a course to the database. If the course id is provided,
225         the method updates and existing course, otherwise, a new course
226         is created.
227 
228         :author: Tobias Huber, Claas Voelcker
229         :param request: request containig the user and auth token
230         :param course_id: optional: the course id
231                     (if a course is edited instead of created)
232         :param format: unused (inherited)
233         :return: a status response giving feedback about errors or a sucessful
234                     database access to the frontend
235         """
236 
237         data = request.data
238 
239         # checks whether the request contains any data
240         if data is None:
241             return Response({'error': 'Request does not contain data'},
242                             status=status.HTTP_400_BAD_REQUEST)
243 
244         course_id = data.get('id')
245         # Checks whether the name of the new course is unique
246         if (course_id is None) and Course.objects.filter(
247                 name=data['name']).exists():
248             return Response({'error': 'Course with that name exists'},
249                             status=status.HTTP_409_CONFLICT)
250 
251         # adds the user of the request to the data
252         if course_id is None:
253             data['responsible_mod'] = request.user
254         # if the course is edited, check for editing permission
255         else:
256             responsible_mod = Course.objects.get(id=course_id).responsible_mod
257             # decline access if user is neither admin nor the responsible mod
```

```
258             if (request.user.profile.is_admin()
259                     or request.user == responsible_mod):
260                 data['responsible_mod'] = responsible_mod
261             else:
262                 raise PermissionDenied(detail="You're not allowed to edit this"
263                                        + "course, since you're not the"
264                                        + 'responsible mod',
265                                        code=None)
266
267         # serialize the course
268         course_serializer = serializers.CourseSerializer(data=data)
269
270         # check for serialization errors
271         if not course_serializer.is_valid():
272             return Response({'error': course_serializer.errors},
273                             status=status.HTTP_400_BAD_REQUEST)
274
275         # send the data to the frontend
276         else:
277             try:
278                 course_serializer.create(data)
279                 return Response({'success': 'Course saved'},
280                                 status=status.HTTP_201_CREATED)
281             except ParseError as error:
282                 return Response({'error': str(error)},
283                                 status=status.HTTP_400_BAD_REQUEST)
284
285
286 class ToggleCourseVisibilityView(APIView):
287     """
288     changes the visibility of a course
289
290     alternatively sets the visibility to the provided state
291     {
292         "is_visible": (optional) True|False
293     }
294
295     @author Tobias Huber
296     """
297
298     authentication_classes = (authentication.TokenAuthentication,)
299     permission_classes = (custom_permissions.IsAdmin,)
300
301     def post(self, request, course_id):
302         """
303         Sets the course visibility to the given value
304         :param request: request object from the rest dispatcher
305         :param course_id: the id of the course whos visibility shall be changed
306         :return: a REST Response with a meaningfull JSON formatted message
307         """
308         if course_id is None:
309             return Response({'ans': 'course_id must be provided'},
310                             status=status.HTTP_400_BAD_REQUEST)
311         elif not Course.objects.filter(id=course_id).exists():
312             return Response({'ans': 'course not found. id: ' + course_id},
313                             status=status.HTTP_404_NOT_FOUND)
314         else:
315             course = Course.objects.get(id=course_id)
316             if 'is_visible' in request.data:
317                 if not (request.data['is_visible'] == 'true'
318                         or request.data['is_visible'] == 'false'):
319                     Response({'ans':'is_visible must be "true" or "false" of type string'}
320                 course.is_visible = request.data['is_visible'] == 'true'
321             else:
322                 course.is_visible = not course.is_visible
323             course.save()
```

```
324             return Response({'is_visible': course.is_visible},
325                             status=status.HTTP_200_OK)
326
327
328 class ModuleView(APIView):
329     """
330     Shows a module
331     @author Claas Voelcker
332     """
333     authentication_classes = (authentication.TokenAuthentication,)
334     permission_classes = (permissions.IsAuthenticated,)
335
336     def get(self, request, course_id, module_id, format=None):
337         """
338         Not implemented
339         """
340         return Response({'ans': 'Method not allowed'},
341                         status=status.HTTP_405_METHOD_NOT_ALLOWED)
342
343     def post(self, request, format=None):
344         """
345         Not implemented
346         """
347         return Response({'ans': 'Method not allowed'},
348                         status=status.HTTP_405_METHOD_NOT_ALLOWED)
349
350
351 class QuestionView(APIView):
352     """
353     View to show questions and to evaluate them. This does not return the
354     answers, which are given by a separate class.
355     @author Claas Voelcker
356     """
357     authentication_classes = (authentication.TokenAuthentication,)
358     permission_classes = (permissions.IsAuthenticated,)
359
360     @staticmethod
361     def can_access_question(user, question, module_id, question_id):
362         """
363         Checks if the question is accessable by the user (all questions before
364         need to be answered correctly)
365         :param user: user wanting to access
366         :param question: question to be accessed
367         :param module_id: module id the question belongs to
368         :param question_id: the questions id
369         :return: True|False (see description)
370         @author Tobias Huber
371         """
372         module = question.module
373         first_question = int(module_id) <= 0 and int(question_id) <= 0
374         if first_question:
375             return True
376         elif (not first_question
377               and question.get_previous_in_order()
378               and Try.objects.filter(
379                   user=user,
380                   question=question.get_previous_in_order(),
381                   solved=True)):
382             return True
383         elif (not module.is_first_module()
384               and module.get_previous_in_order()
385               and Try.objects.filter(
386                   user=user,
387                   question=module.get_previous_in_order().question_set.all()[0],
388                   solved=True)):
389             return True
```

```
390         return False
391
392     def get(self, request, course_id, module_id, question_id, format=None):
393         """
394         Get a question together with additional information about the module
395         and position (last_module and last_question keys)
396         """
397         try:
398             course = Course.objects.get(id=course_id)
399             course_module = course.module_set.all()[int(module_id)]
400             question = course_module.question_set.all()[int(question_id)]
401
402             if question is None:
403                 return Response({'ans': 'Question not found'},
404                                 status=status.HTTP_404_NOT_FOUND)
405             if not self.can_access_question(request.user, question, module_id,
406                                             question_id):
407                 return Response({'ans': "Previous question(s) haven't been "
408                                         'answered correctly yet'},
409                                 status=status.HTTP_403_FORBIDDEN)
410             data = serializers.QuestionSerializer(question,
411                                                   context={'request': request})
412             data = data.data
413             return Response(data, status=status.HTTP_200_OK)
414         except Exception as error:
415             return Response({'error': str(error)},
416                             status=status.HTTP_404_NOT_FOUND)
417
418     def post(self, request, course_id, module_id, question_id, format=None):
419         """
420         Evaluates the answer to a question.
421         @author Tobias Huber
422         """
423         try:
424             course = Course.objects.get(id=course_id)
425             course_module = course.module_set.all()[int(module_id)]
426             question = course_module.question_set.all()[int(question_id)]
427         except Exception:
428             return Response({'ans': 'Question not found'},
429                             status=status.HTTP_404_NOT_FOUND)
430         # deny access if there is a/are previous question(s) and it/they
431         # haven't been answered correctly
432         if not (self.can_access_question(request.user, question, module_id,
433                                          question_id)):
434             return Response(
435                 {'ans': "Previous question(s) haven't been answered"
436                         + " correctly yet"},
437                 status=status.HTTP_403_FORBIDDEN
438             )
439
440         solved = question.evaluate(request.data["answers"])
441
442         # only saves the points if the question hasn't been answered yet
443         if solved and not question.try_set.filter(
444                 user=request.user, solved=True).exists():
445             request.user.profile.ranking += question.get_points()
446             request.user.profile.save()
447         Try(user=request.user, question=question,
448             answer=str(request.data["answers"]), solved=solved).save()
449         response = {"evaluate": solved}
450         if solved:
451             next_type = ""
452             if not question.is_last_question():
453                 next_type = "question"
454             elif not course_module.is_last_module():
455                 next_type = "module"
```

```
456             elif course.quizquestion_set.exists():
457                 next_type = "quiz"
458             response['next'] = next_type
459             if solved and question.feedback:
460                 # response['custom_feedback'] = question.custom_feedback()
461                 response['feedback'] = question.feedback
462         return Response(response)
463
464
465 class AnswerView(APIView):
466     """
467     Shows all possible answers to a question.
468     :author: Claas Voelcker
469     """
470     authentication_classes = (authentication.TokenAuthentication,)
471     permission_classes = (permissions.IsAuthenticated,)
472
473     def get(self, request, course_id, module_id, question_id, format=None):
474         """
475         Lists the answers for a question
476         """
477         course = Course.objects.get(id=course_id)
478         module = course.module_set.all()[int(module_id)]
479         question = module.question_set.all()[int(question_id)]
480         answers = question.answer_set()
481         data = [serializers.get_answer_serializer(answer) for answer in
482                 answers]
483         return Response(data, status=status.HTTP_200_OK)
484
485     def post(self, request, format=None):
486         """
487         Not implemented
488         :param request:
489         :param format:
490         :return:
491         """
492         return Response({"ans": 'Method not allowed'},
493                         status=status.HTTP_405_METHOD_NOT_ALLOWED)
494
495
496 def calculate_quiz_points(old_percentage, new_percentage, difficulty):
497     """
498     calculates the quiz points from the old existing statistics and the new
499     quiz answers
500     :param old_percentage: percentage of questions already answered correctly
501     :param new_percentage: percentage of questions newly answered correctly
502     :param difficulty: the course difficulty
503     :return: the additional points
504     """
505     multiplier = 2 if difficulty == 2 else 1
506     ranking_threshold = [0.4, 0.7, 0.9]
507     old_extra_points = [x[0] for x in enumerate(ranking_threshold) if
508                         x[1] > old_percentage]
509     new_extra_points = [x[0] for x in enumerate(ranking_threshold) if
510                         x[1] > new_percentage]
511     old_extra_points = 3 if not old_extra_points else old_extra_points[0]
512     new_extra_points = 3 if not new_extra_points else new_extra_points[0]
513     return max(0, 5 * multiplier * (new_extra_points - old_extra_points))
514
515
516 class QuizView(APIView):
517     """
518     Shows the quiz question of the current course in get
519     evaluates this quiz question in post
520     @author Leonhard Wiedmann
521     """
```

```
522     authentication_classes = (authentication.TokenAuthentication,)
523     permission_classes = (permissions.IsAuthenticated,)
524
525     def get(self, request, course_id):
526         """
527         Shows the current quiz question if it exists.
528         When this id does not exist throws error message
529         """
530         course = Course.objects.filter(id=course_id).first()
531
532         # check if user did last question of the last module
533         # if valid the course is completed
534         module = course.module_set.all()[len(course.module_set.all()) - 1]
535         question = module.question_set.all(
536         )[len(module.question_set.all()) - 1]
537         if not Try.objects.filter(question=question, solved=True).exists():
538             return Response({"error": "complete the course first"},
539                             status=status.HTTP_403_FORBIDDEN)
540
541         quiz = course.quizquestion_set.all()
542         if len(quiz) in range(5, 21):
543             quiz = serializers.QuizSerializer(quiz, many=True)
544
545             return Response(quiz.data)
546         return Response({"error": "this quiz is invalid"},
547                         status=status.HTTP_400_BAD_REQUEST)
548
549     def post(self, request, course_id, format=None):
550         """
551         Resolves this quiz question for the current user.
552         """
553         # post does two things (return feedback for questions and whole quiz)
554         # this switch/case differentiates between the two
555         if request.data['type'] == "check_answers":
556             course = Course.objects.get(id=course_id)
557             quiz = course.quizquestion_set.all()
558             all_question_length = len(quiz)
559             if all_question_length <= 0:
560                 return Response({"error": "this quiz does not exist"},
561                                 status=status.HTTP_404_NOT_FOUND)
562             # checks if the submission is wrong (different lengths of the
563             # arrays)
564             if len(quiz) != len(request.data['answers']):
565                 resp = "the quiz has {} question and your evaluation has {}"\
566                     .format(len(quiz), len(request.data['answers']))
567                 return Response({"error": resp, "test": request.data},
568                                 status=status.HTTP_400_BAD_REQUEST)
569             response = []
570             newly_solved = 0
571             old_solved = 0
572             for i, quiz_entry in enumerate(quiz):
573                 answer_solved = request.data['answers'][i]
574                 for answer in request.data['answers']:
575                     if 'id' in answer and quiz_entry.id is answer['id']:
576                         answer.pop('id')
577                         answer_solved = answer
578                         break
579                 solved = quiz_entry.evaluate(answer_solved)
580                 if solved and not quiz_entry.try_set.filter(
581                         user=request.user, solved=True).exists():
582                     newly_solved += 1
583                     request.user.profile.ranking += quiz_entry.get_points()
584                 elif quiz_entry.try_set.filter(user=request.user,
585                                                solved=True).exists():
586                     old_solved += 1
587                 Try(user=request.user, quiz_question=quiz_entry,
```

```
588                     answer=str(request.data), solved=solved).save()
589
590                 response.append({"name": quiz[i].question, "solved": solved})
591
592             old_extra = float(old_solved / all_question_length)
593             new_extra = float(
594                 (newly_solved + old_solved) / all_question_length)
595             request.user.profile.ranking += calculate_quiz_points(
596                 old_extra, new_extra, course.difficulty)
597             request.user.profile.save()
598
599             return Response(response, status=status.HTTP_200_OK)
600         if request.data['type'] == 'get_answers':
601             course = Course.objects.get(id=course_id)
602             quiz_question = course.quizquestion_set.all()[request.data['id']]
603             answers = [answer.id for answer in
604                        quiz_question.quizanswer_set.all() if answer.correct]
605             return Response({'answers': answers}, status.HTTP_200_OK)
606         return Response({'ans': 'Could not process request'},
607                         status.HTTP_400_BAD_REQUEST)
608
609
610 class UserView(APIView):
611     """
612     Shows a user profile
613     @author Claas Voelcker
614     """
615     authentication_classes = (authentication.TokenAuthentication,)
616     permission_classes = (permissions.IsAuthenticated,)
617
618     def get(self, request, user_id=False, format=None):
619         """
620         Shows the profile of any user if the requester is mod,
621         or the profile of the requester
622
623         TODO: If the behaviour that an admin is allowed to receive information
624         about a specific user, will be used again,
625         a custom_permission should be written.
626         """
627         user = request.user
628         if user_id:
629             if user.profile.is_admin():
630                 user = User.objects.filter(id=user_id).first()
631                 if not user:
632                     return Response({'ans': 'User not found'},
633                                     status=status.HTTP_404_NOT_FOUND)
634             else:
635                 raise PermissionDenied(detail=None, code=None)
636
637         user = serializers.UserSerializer(user)
638         return Response(user.data)
639
640     def post(self, request, format=None):
641         """
642         Post is used to update the profile of the requesting user
643         @author Tobias Huber
644         """
645         user = request.user
646         data = request.data
647
648         if 'oldpassword' in data:
649             if not request.user.check_password(request.data['oldpassword']):
650                 return Response({'error': 'given password is incorrect'},
651                                 status=status.HTTP_400_BAD_REQUEST)
652         else:
653             return Response({'error': 'password is required'},
```

```
654                             status=status.HTTP_400_BAD_REQUEST)
655
656         user_serializer = serializers.UserSerializer(user, data=data,
657                                                      partial=True)
658         if user_serializer.is_valid():
659             user_serializer = user_serializer.update(
660                 user,
661                 validated_data=request.data)
662             return Response({'ans': 'Updated user ' + user.username},
663                             status=status.HTTP_200_OK)
664         return Response(user_serializer.errors,
665                         status=status.HTTP_400_BAD_REQUEST)
666
667
668 class UserRegisterView(APIView):
669     """
670     Saves a new user
671     @author Tobias Huber
672     """
673     authentication_classes = []
674     permission_classes = []
675
676     def post(self, request, user_id=False, format=None):
677         """
678         Saves a new user.
679         """
680         if user_id:
681             return Response({'ans': 'Please use the UserView to update data'},
682                             status=status.HTTP_403_FORBIDDEN)
683         user_serializer = serializers.UserSerializer(data=request.data)
684         if user_serializer.is_valid():
685             user_serializer.create(request.data)
686             return Response({'ans': 'Created a new user'},
687                             status=status.HTTP_201_CREATED)
688         return Response(user_serializer.errors,
689                         status=status.HTTP_400_BAD_REQUEST)
690
691
692 class MultiUserView(APIView):
693     """
694     Shows an overview over all users
695     @author Claas Voelcker
696     """
697     authentication_classes = (authentication.TokenAuthentication,)
698     permission_classes = (custom_permissions.IsAdmin,)
699
700     def get(self, request):
701         """
702         Returns all users
703         """
704         users = User.objects.all()
705         data = serializers.UserSerializer(users, many=True).data
706         return Response(data)
707
708     def post(self, request, format=None):
709         """
710         Not implemented
711         """
712         return Response({'ans': 'Method not allowed'},
713                         status=status.HTTP_405_METHOD_NOT_ALLOWED)
714
715
716 class StatisticsView(APIView):
717     """
718     A class displaying statistics information for a given user. It is used to
719     access the try object.
```

```
720     @author: Claas Voelcker
721     """
722     authentication_classes = (authentication.TokenAuthentication,)
723     permission_classes = (permissions.IsAuthenticated,)
724
725     def get(self, request, user_id=None):
726         """
727         shows the statistics of the given user
728         """
729         user = request.user if not user_id else User.objects.get(id=user_id)
730         tries = Try.objects.filter(user=user)
731         data = serializers.TrySerializer(tries, many=True).data
732         return Response(data)
733
734     def post(self, request, format=None):
735         """
736         implements filtering logic for the statistics
737         :param request:
738         :param format:
739         :return:
740         """
741         import time
742         import csv
743         data = request.data
744         user = request.user
745
746         tries = Try.objects.all()
747
748         groups = user.groups.values_list('name', flat=True)
749
750         is_mod = 'moderator' in groups or 'admin' in groups
751
752         # the simplest call is if the user just wants its statistic
753         if 'id' in data and data['id'] == user.id:
754             tries = tries.filter(user=user)
755
756         # A moderator can get all statistics of his created courses
757         # with 'get_courses' as in put the it will return all courses created
758         # by this user
759         elif is_mod and 'course' in data and 'admin' not in groups:
760             tries = tries.filter(
761                 question__module__course__responsible_mod=user)
762
763         # admins can get all statistics of all users
764         elif 'admin' in groups:
765             if 'id' in data:
766                 tries.filter(user__id=data['id'])
767         else:
768             return Response({'error': 'invalid userID'},
769                             status=status.HTTP_403_FORBIDDEN)
770
771         # return all statistics after prefiltering for this course
772         if 'course' in data:
773
774             tries = tries.filter(question__module__course__id=data['course'])
775
776             if 'list_questions' in data:
777                 course = Course.objects.filter(id=data['course']).first()
778                 value = []
779                 index = 0
780                 for module in course.module_set.all():
781                     value.append([])
782                     for question in module.question_set.all():
783                         value[index].append({'name': question.title,
784                                              'solved': len(
785                                                  question.try_set.filter(
```

```
786                                                     solved=True).all()),
787                                             'not solved': len(
788                                                 question.try_set.filter(
789                                                     solved=False).all())})
790                     index += 1
791                 return Response(value)
792
793         # get the statistics for a specific time
794         if ('date' in data
795                 and 'start' in data['date']
796                 and 'end' in data['date']):
797             start = data['date']['start']
798             end = data['date']['end']
799             tries = tries.filter(
800                 date__range=[start, end])
801
802         # filter just for solved tries
803         if 'solved' in data:
804             tries = tries.filter(solved=data['solved'])
805
806         # filter for a specific category
807         if 'category' in data:
808             tries = tries.filter(
809                 question__module__course__category__name=data['category'])
810
811         # if this variable is set the view will return a array of dicts which
812         # are {name: string, color: string, counter: number}
813         if 'categories__with__counter' in data:
814             categories = CourseCategory.objects.all()
815             value = []
816             for cat in categories:
817                 value.append(
818                     {
819                         'name': cat.name,
820                         'color': cat.color,
821                         'counter': len(tries.filter(
822                             question__module__course__category=cat))
823                     })
824             return Response(value)
825
826         serialize_data = None
827
828         # filters the statistics and counts for the 'filter' variable
829         if 'filter' in data:
830             value = {}
831             for trie in tries:
832                 if not str(getattr(trie, data['filter'])) in value:
833                     value[str(getattr(trie, data['filter']))] = 1
834                 else:
835                     value[str(getattr(trie, data['filter']))] += 1
836             return Response(value)
837
838         # this part orders the list for the 'order' value in the request
839         if 'order' in data:
840             tries = tries.order_by(data['order'])
841
842         if 'serialize' in data:
843             serialize_data = serializers.TrySerializer(tries, many=True,
844                                                        context={
845                                                            'serialize': data[
846                                                                'serialize']}).data
847         else:
848             serialize_data = serializers.TrySerializer(tries, many=True).data
849
850         if 'format' in data and data['format'] == 'csv':
851             response = HttpResponse(content_type='text/csv')
```

```
852             filename = time.strftime('%d/%m/%Y') + '-' + user.username + '.csv'
853             content = 'attachment; filename="' + filename
854             response['Content-Disposition'] = content
855             writer = csv.writer(response)
856             writer.writerow(['question', 'user', 'date', 'solved'])
857             for row in serialize_data:
858                 profile = Profile.objects.get(user__username=row['user'])
859                 profile_hash = profile.get_hash()
860                 writer.writerow(
861                     [row['question'],
862                      profile_hash,
863                      row['date'],
864                      row['solved']])
865             return response
866         return Response(serialize_data)
867
868
869 class RankingView(APIView):
870     """
871     A view for the ranking. The get method returns an ordered list of all users
872     according to their rank.
873     """
874     authentication_classes = (authentication.TokenAuthentication,)
875     permission_classes = (permissions.IsAuthenticated,)
876
877     def get(self, request, format=None):
878         """
879         API request for ranking information
880         :param request: can be empty
881         :param format: request: can be empty
882         :return: a json response with ranking information
883         """
884         profiles = Profile.objects.all().reverse()
885         data = serializers.RankingSerializer(profiles).data
886         return Response(data)
887
888     def post(self, request, format=None):
889         """
890         Not implemented
891         """
892         return Response({'ans': 'Method not allowed'},
893                         status=status.HTTP_405_METHOD_NOT_ALLOWED)
894
895
896 class RequestView(APIView):
897     """
898     The RequestView class is used to submit a request for moderator rights.
899
900     The request can be accessed via "clonecademy/user/request/"
901     @author Tobias Huber
902     """
903     authentication_classes = (authentication.TokenAuthentication,)
904     permission_classes = (permissions.IsAuthenticated,)
905
906     def get(self, request, format=None):
907         """
908         Returns True if request is allowed and False if request isn't allowed
909         or the user is already mod.
910         """
911         allowed = (not request.user.profile.is_mod()
912                    and request.user.profile.modrequest_allowed())
913         return Response({'allowed': allowed},
914                         status=status.HTTP_200_OK)
915
916     def post(self, request, format=None):
917         """
```

```
        Handels the moderator rights request. Expects a reason and extracts the
        user from the request header.
        """
        data = request.data
        user = request.user
        profile = user.profile
        if not user.profile.modrequest_allowed():
            return Response(
                {'ans': 'User is mod or has sent too many requests'},
                status=status.HTTP_403_FORBIDDEN)
        # pay attention because there could be localization errors
        profile.last_modrequest = timezone.now()
        profile.save()
        send_mail(
            'Moderator rights requested by {}'.format(user.username),
            'The following user {} requested moderator rights for the'
            'CloneCademy platform. \n'
            'The given reason for this request: \n{}\n '
            'If you want to add this user to the moderator group, access the '
            'profile {} for the confirmation field.\n '
            'Have a nice day,\n your CloneCademy bot'.format(
                user.username, data['reason'],
                user.profile.get_link_to_profile()),
            'bot@clonecademy.de',
            [admin.email for
             admin in Group.objects.get(name='admin').user_set.all()]
        )
        return Response({'Request': 'ok'}, status=status.HTTP_200_OK)


class UserRightsView(APIView):
    """
    Used to promote or demote a given user (by id)

    This View is used to grant or revoke specific rights (user|moderator|admin)
    The POST data must include the following fields
    {"right": "moderator"|"admin",
    "action": "promote"|"demote"}.
    Returns the request.data if validation failed.

    The user_id is to be provided in the url.

    TODO: try the generic.create APIView. Its behaviour isn't really different
    from the current. It just provides additional success-headers in a way
    I do not understand.
    """

    authentication_classes = (authentication.TokenAuthentication,)
    permission_classes = (custom_permissions.IsAdmin,)

    def post(self, request, user_id, format=None):
        """
        changes the group membership of the user
        """
        data = request.data
        right_choices = ['moderator', 'admin']
        action_choices = ['promote', 'demote']
        errors = {}

        # validation
        if not data['right'] or not data['right'] in right_choices:
            errors['right'] = ('this field is required and must be one of '
                               + 'the following options'
                               + ', '.join(right_choices))
        if not data['action'] or not data['action'] in action_choices:
            errors['action'] = ('this field is required and must be one of '
```

```
984                                 + 'the following options'
985                                 + ', '.join(action_choices))
986         if not User.objects.filter(id=user_id).exists():
987             errors['id'] = 'a user with the id #' + user_id + ' does not exist'
988         if errors:
989             return Response(errors, status=status.HTTP_400_BAD_REQUEST)
990
991         # actual behaviour
992         user = User.objects.get(id=user_id)
993         group = Group.objects.get(name=data['right'])
994         action = data['action']
995         if action == 'promote':
996             user.groups.add(group)
997         elif action == 'demote':
998             user.groups.remove(group)
999         return Response(serializers.UserSerializer(user).data)
1000
1001     def get(self, request, user_id, format=None):
1002         """
1003         This API is for debug only.
1004         It comes in quite handy with the browsable API
1005         """
1006         user = User.objects.get(id=user_id)
1007         return Response({'username': user.username,
1008                          'is_mod?':
1009                              user.groups.filter(name='moderator').exists(),
1010                          'is_admin?':
1011                              user.groups.filter(name='admin').exists()})
1012
1013
1014 class PwResetView(APIView):
1015     """
1016     Resets the password of a user and sends the new one to the email adress
1017     of the user
1018
1019     {
1020         "email": the email of the user
1021     }
1022     """
1023
1024     authentication_classes = ()
1025     permission_classes = ()
1026
1027     def post(self, request, format=None):
1028         """
1029         Sends a mail to the user containing a new one time password
1030         :param request:
1031         :param format:
1032         :return:
1033         """
1034         data = request.data
1035         if 'email' not in data:
1036             return Response({'ans': 'you must provide an email'},
1037                             status=status.HTTP_400_BAD_REQUEST)
1038         elif not User.objects.filter(email=data['email']).exists():
1039             return Response({'ans': 'no user with email: ' + data['email']},
1040                             status=status.HTTP_404_NOT_FOUND)
1041
1042         # if request data is valid:
1043         user = User.objects.get(email=data['email'])
1044         # generate a random password with the rand() implementation of
1045         # django.utils.crypto
1046         new_password = get_random_string(length=16)
1047
1048         send_mail(
1049             'Password Reset on clonecademy.net',
```

```
1050            ('Hello {},\n \n'
1051             + 'You have requested a new password on clonecademy.net \n'
1052             + 'Your new password is: \n{} \n \n'
1053             + 'Please change it imediately! \n'
1054             + 'Have a nice day,\nyour CloneCademy bot').format(
1055                 user.username, new_password),
1056            'bot@clonecademy.de',
1057            [user.email]
1058        )
1059        user.set_password(new_password)
1060        user.save()
1061        return Response(status=status.HTTP_200_OK)
```

« index coverage.py v4.4.1, created at 2017-09-28 15:34

G GitHub-Wiki

Das Team hat die Dokumentation der REST-API wie geplant im GitHub Wiki des Repositories geführt. Um die Webseiten lesbar für dieses Dokument zu machen, haben wir die Markdown Seiten der WIki in LaTeXCode umgewandelt. Im folgenden sind die einzelnen Seiten aufgeführt.

UserView

Introduction

The UserView class is used to view information about a single user. The ‘post’ method is used to register new users

The request can be accessed via “clonecademy/user/”

The url “clonecademy/user/” Points to the user itself.

Method Details

‘get’

Permissions: mod or higher for other, user for self

Header: user id (for permissions)

Response:

```
{
"id": number id,
"username": sting "The users username",
"email": string "the users email address",
"first_name": string "First name of the user (if provided, else empty)",
"last_name": string "First name of the user (if provided, else empty)",
"groups": [string] "A array of strings with the names of the group. E.g ['moderator',
"date_joined": string "The date on which this user was created",
"language": string "short form of the language, 'eg' for english"
"avatar": string "encoded image of this user",
"ranking": number "the ranking of the current user"
}
```

‘post’

Header:

Request:

```
{
  "user_id": number id;
  "username": string "The users username";
  "email": string "the users email address";
  "password": string "the chosen password of the user";
  ("first_name": string "First name of the user";)
  ("last_name": string "First name of the user";)
  ("age": string "birthday of the user")
}
```

Response: 200 (user saved), 500 + serializer error (error)

Course View

Introduction

The CourseView class is used to get information for a specific course. Courses are referenced via the general url “clonecademy.com/courses/”.

Method Details

‘get’

Header: user id (for permissions), course id (from url)

Response:

```
{
"name": string "Course name";
"difficulty": number 0/1/2/3;
"category": string "iGEM/General";
"language": string "de/en";
"modules": [{
              "name": string "Module name";
              "learning_text": string "Sample learning text";
              "questions": [{
                              "type": string "MultipleChoiceQuestion";
                              "solved": boolean True/False;
                            }];
            }];
}
```

‘post’

Header: user id (for permissions)

Request:

```
{
"name": string "Course name",
"difficulty": number 0/1/2/3,
```

```
"modules": [{
              "name": string "Module name",
              "learning_text": string "Sample learning text";
              "questions": [{
                               "type": string "MultipleChoiceQuestion";
                               "question_title": string "(currently blank)";
                               "question_body": string "The text for the question";
                               "feedback": string "A specific feedback text (leave empt
                            }]
            }]
"quiz": [{
        "question" : string,
        "image": string,
        "answers": [{
                    "text": string,
                    "img": string,
                    "correct": boolean,
                    }]
         }]
}
```

Response: 201 (saved), 500 + serializer error (error)

GrantModRightsView

Introduction

The GrantModRightsView class is used to promote a user to a moderator or in other words add him*her to the usergroup 'moderator'. It also provides a get method to show if a user is already moderator

Method Details

'get'

URL: user_id

Returns true, when the user in the url is a moderator

Request

Response

```
True|False (200)
```

'post'

Header: user_id Request:

```
{ }
```

Note: May be user for other permissions in the future

Response:

```
"successfully promoted " + to_be_promoted.username (200)
"the user \" "+ to_be_promoted.username +"\" is already a moderator" (200)
```

MultiCourseView

Introduction

The MultiCourseView class is used to get information for all course. The course list can be view by calling “clonecademy.com/courses/list”

Method Details

‘get’

Not implemented

‘post’

Header: user id (for permissions)

Request:

```
{
"type": string "all/mod/started"
"category": string "iGEM/General/all"
"language": string "de/en"
}
```

Response:

```
[{
"id": string "the courses id",
"name": string "the courses name",
"category": string "iGEM/...",
"num_questions": int "amount of questions in the course",
"num_answered": int "amount of questions the user answered",
"responsible_mod": int "the id of the responsive moderator for this course",
}]
```

MultiUserView

Introduction

The MultiUserView class is used to view information about all users.

The request can be accessed via “clonecademy/user/all”

Method Details

‘get’

Permissions: mod or higher

Header: user id (for permissions)

Response: [{ “user_id”: “id”; “username”: “The users username”; }]

‘post’

Not implemented yet (could be used for filtering the input list)

QuestionView

Introduction

The QuestionView class is used to provide information about every question, provided the user is allowed to see the question. The class is also used to answer questions.

Method Detais

‘get’

Header: user id (for permissions), course id (from url), ordered position of module in coures (from url), ordered position of question in module (from url)

Response:

```
{
"title": string "Question title (currently from module)"
"body": string "Question text"
"feedback": string "Feedback for user after correct answer"
"last": boolean "True/False"
"module": string "the corresponding module for the question"
"last_module": boolean "True/False"
"learning_text": string "The modules learning text"
"course": string "the corresponding course"
"progress": [[string]] "contains the title of all questions of this course"
}
```

Error:

Response: 400 if unauthorized // only allowed for user if it is the first in this course or if the user answered the questions before correct

‘post’

Header: user id (for permissions), course id (from url), ordered position of module in coures (from url), ordered position of question in module (from url)

Request:

```
{
"answers":depends on the question type
}
```

Response:

```
{
"solved": boolean true/false;
"next": string "question/module/quiz for the next element"
### potentially more in a future implementation
}
```

Response: 200 (correct answer, try saved), 400 unauthorized, 500+ serializer error

Error:

Response: 400 if unauthorized // only allowed for user if it is the first in this course or if the user answered the questions before correct

QuizView

Introduction

The QuizView class is used to provide information about every quiz question, provided the user is allowed to see the quiz question. The class is also used to answer quiz questions.

Method Detais

‘get’

Header: user id (for permissions), course id (from url), ordered position of module in coures (from url), ordered position of question in module (from url)

Response:

```
{
"id": number,
"question": string,
"imgage": string,
"answers": [{
           "id": number,
           "text": string,
           "img": string
           }]
}
```

Error:

Response: 400 if unauthorized // only allowed for user if it is the first in this course or if the user answered the questions before correct

‘post’

Header: user id (for permissions), course id (from url)

Request:

```
[{
```

```
 "id": number "the quiz question id"
 "answers": [{
            "chosen": boolean "if true the user has checked this quiz question"
            "id": number "the id of the quiz answer"
            }]
}]
```

Response:

```
[{
"name": string "the title of the quiz question",
"solved": boolean "if the answer was correct"
}]
```

Response: 200 (correct answer, try saved), 400 unauthorized, 500+ serializer error

Error:

Response: 400 if unauthorized // only allowed for user if it is the first in this course or if the user answered the questions before correct

RequestView

Introduction

The RequestView class is used to submit a request for moderator rights.

The request can be accessed via “clonecademy/user/request/”

Method Details

‘get’

Header: user id (for permissions and filtering)

Response: 200 (can submit request), 403 + reason (is mod or to many mod requests)

‘post’

Header: user id (for permissions and saving)

Request:

```
{
"reason": string "the reason for the request",
}
```

Response: 200 (request sent), 500 + serializer error (error)

Side effects: sends an email to the admins requesting mod access

StaticPages

Static pages

All static pages are html Pages and have to be in the folder **assets/statics** To find the Page on the menu you have to add it in the **menu.componenent.ts** in the variable **links**.

```
{ url: (string) has to be the name of the file without html, name: (string) will
be the name in the dropdown menu }
```

The order of the objects in the link variable is the order of the dropdown menu.

StatisticsView

Introduction

The StatisticsView class is used to get information about the usage statistics. The get method returns all Try objects, detailing all questions and the respective answers, the post method saves a new Try object, when an answer is submitted.

Statistics can be accessed via “clonecademy/statistics/”

Method Details

‘get’

Header: user id (for permissions and filtering)

Response:

```
[{
"question": string "the questions id";
"module": string "The questions module"
"course": string "The questions course"
"solved": boolean True/False;
"date": string "The date of the try"
}]
```

‘post’

Header: user id (for permissions and filtering)

Post:

```
{
 "format": string "valid formats are ’csv’ or empty for json";
 "id": string "the user ID if the user is not a moderator or a admin this must be set;
 "get_courses": any "if this field is set and the current user is a moderator or admin
"course": number "get the statistics for this course (with prefiltering admin moderato
"date": {
          "start": string "the start date for the statistics it has to be the format ’
```

```
          "end": string "the end date for the statistics '%Y-%m-%d %H:%M:%S'";
        },
"serialize": string "get more fields of the statistics for example question__module__c
"order": string "order the tries",
"filter": string "filter the statistics for this class and count the amounts"
}
```

Response:

```
[{
"question": string "the questions id";
"user": string "The User for this try";
"solved": boolean True/False;
"date": string "The date of the try"
}]
```

ToggleCourseVisibilityView

Introduction

This class is used to change the “is_visible” state of a course. ‘True’ means, that users can see the course in their dashboards and False the opposite. Courses are referenced via the general url “clonecademy.com/courses/”.

Method Details

‘post’

Header: user id (for permissions), course id (from url)

Request: Leaving out the “is_visible” content of the request, and therefore posting no data at all, toggles the state.

```
{
"is_visible": True|False (optional)
}
```

Response: 200 (visibility changed), 400-404 (Bad Requests), 500+ (internal error)

UserRightsView

Introduction

The UserRightsView class is used to promote or demote a user to Moderator or Admin.

Method Details

‘get’

No get Method implemented

‘post’

Header: user_id Request:

```
{
’right’: string "the group in which the user will be pro or demoted",
’action’: string "promote or demote",
}
```

Response:

The response is identical to the UserView get method

UserView

Introduction

The UserView class is used to view information about a single user. The ‘post’ method is used to register new users

The request can be accessed via “clonecademy/user/”

The url “clonecademy/user/” Points to the user itself.

Method Details

‘get’

Permissions: mod or higher for other, user for self

Header: user id (for permissions)

Response:

```
{
"id": number id,
"username": sting "The users username",
"email": string "the users email address",
"first_name": string "First name of the user (if provided, else empty)",
"last_name": string "First name of the user (if provided, else empty)",
"groups": [string] "A array of strings with the names of the group. E.g ['moderator',
"date_joined": string "The date on which this user was created",
"language": string "short form of the language, 'eg' for english"
"avatar": string "encoded image of this user",
"ranking": number "the ranking of the current user"
}
```

‘post’

Header:

Request:

```
{
  "user_id": number id;
  "username": string "The users username";
  "email": string "the users email address";
  "password": string "the chosen password of the user";
  ("first_name": string "First name of the user";)
  ("last_name": string "First name of the user";)
  ("age": string "birthday of the user")
}
```

Response: 200 (user saved), 500 + serializer error (error)

H Userstories

ID	2
Benutzerrolle	Nutzer*in
Name	Authentifizierung
Beschreibung	Als Nutzer*in möchte ich mich auf der Seite als Nutzer einloggen können.
Akzeptanzkriterium	Das Authentifizerungsfenster wird korrekt angezeigt. Bei gültiger Eingabe seiner Daten wird der*die Nutzer*in auf die Kursübersicht weitergeleitet und ein Authentifizierungstoken wird generiert, der zur Authetifizierung weitergereicht wird. Bei ungültiger Eingabe wird der*die Nutzer*in durch ein Popup darauf hingewiesen und nicht eingelogt.
Story Points	2
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	1
Tatsächlicher Aufwand	3.5 h
Velocity	1.75 h/SP
Bemerkung	

ID	3
Benutzerrolle	Moderator*in
Name	Neuen Kurs anlegen
Beschreibung	Als Moderator*in möchte ich einen neuen Kurs anlege können. Dazu gehe ich auf die Kursübersicht und wähle die Schaltfläche ”Add Course” aus. Ein Editor erlaubt mir, Name und Kategorie des Kurses auszuwählen. Danach kann ich den Kurs speichern und er taucht auf der Übersichtsseite auf. Der Kurs ist erste einmal nur für Moderator*innen sichtbar.
Akzeptanzkriterium	Der neu angelegte Kurs mit Name und Kategorie wird persistent gespeichert und ist für Moderator*innen einsehbar.
Story Points	3
Entwickler	Claas Völcker & Leonhard Wiedmann
Umgesetzt in Iteration	1
Tatsächlicher Aufwand	3 h
Velocity	1.5 h/SP
Bemerkung	Die genaue Funktionsweise des Editors wurde in einem Gespräch mit den Auftraggeber*innen festgelegt.

ID	5
Benutzerrolle	Nutzer*in
Name	Kurs auswählen
Beschreibung	Als Nutzer*in möchte ich aus einer Kursübersicht einen Kurs auswählen und im Folgenden die Fragen des Kurses beantworten können. Nachdem ich den Kurs ausgewählt habe, soll er in meiner persönlichen Kursübersicht samt Fortschrittsangabe angezeigt werden.
Akzeptanzkriterium	Der ausgewählte Kurs wird vollständig angezeigt und die Fragen können bearbeitet und zugleich jederzeit abgebrochen werden.Der Fortschritt wird gespeichert und in einer persönlichen Statistik angezeigt. Nach Abbrechen einer Frage kann die Bearbeitung derselben jederzeit fortgesetzt werden.
Story Points	7
Entwickler	Ilhan Simsiki
Umgesetzt in Iteration	3
Tatsächlicher Aufwand	11 h
Velocity	1.57 h/SP
Bemerkung	

ID	1
Benutzerrolle	Nutzer*inne
Name	Profil erstellen
Beschreibung	Als Nutzer*in möchte ich auf der Seite einen eigenen Account anlegen können. Mit Ausfüllen des Anmeldeformulars mit den Pflichtfeldern: E-Mail-Adresse, Benutzername, Passwort. Und den Optionalen Feldern: Vorname, Nachname, Alter
Akzeptanzkriterium	Das Anmeldeformular wird korrekt dargestellt. Mit dem Klicken des ”Registrieren” Buttons und Ausfüllen der Pflichtfelder wird der Account persistent in der Datenbank gespeichert.
Story Points	8
Entwickler	Tobias Huber
Umgesetzt in Iteration	5
Tatsächlicher Aufwand	20
Velocity	0.4 SP/h
Bemerkung	Der Account wird in der Datenbank persitent gespeichert. Andere Daten werden nicht verändert.

ID	8
Benutzerrolle	Nutzer*in
Name	Frage beantworten
Beschreibung	Ein*e Benutzer*in möchte eine Frage beantworten. Dabei wird jede Frage ausgewertet, und dem*der Nutzer*in wird angezeigt, ob die Eingabe richtig oder falsch war. War die Beantwortung korrekt, wird die nächste Frage angezeigt.
Akzeptanzkriterium	Die Frage kann beantwortet werden und das Ergebnis wird in der Datenbank unter Statistik gespeichert.
Story Points	7 Story Points
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	4
Tatsächlicher Aufwand	14 h
Velocity	2 h/SP
Bemerkung	Es werden neue Daten in die Datenbank geschrieben, jedoch keine existente verändert oder gelöscht.

ID	6
Benutzerrolle	Nutzer*in
Name	Statistik einsehen
Beschreibung	Als Nutzer*in möchte ich einen Button ”Meine Statistik” anklicken können. Nach Anklicken erscheint eine Seite, auf der ich meine bisherigen beantworteten Fragen sehen kann: Wie viele Fragen habe ich richtig beantwortet, wie viele sind falsch.
Akzeptanzkriterium	Die Statistik des*der Nutzer*in wird nach Klicken auf die dafür vorgesehene Schaltfläche vollständig angezeigt.
Story Points	8 Story Points
Entwickler	Claas Völcker
Umgesetzt in Iteration	4
Tatsächlicher Aufwand	8 h
Velocity	1 h/SP
Bemerkung	Es werden keine Daten in der Datenbank geändert.

ID	7
Benutzerrolle	Moderator*in
Name	Multiple Choice Fragen einpflegen
Beschreibung	Als Moderator*in kann ich einem Kurs neue Fragen hinzufügen. Dazu wähle ich auf der Kursübersicht die Schaltfläche neue Frage anlegen aus. Ich kann daraufhin in einem Interface den Name der Frage, den Text der Frage und mögliche Antwortmöglichkeiten eingeben. In einem weiteren Fenster kann ich einen Text eingeben, der den Nutzenden bei erfolgreicher Bearbeitung angezeigt wird und einen Feedbacktext bei falscher Beantwortung.
Akzeptanzkriterium	Eine neu angelegte Frage wird persistent unter dem Kurs gespeichert und ist nach Freigabe für Nutzende bearbeitbar.
Story Points	8 Story Points
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	4
Tatsächlicher Aufwand	15 h
Velocity	1.875 h/SP
Bemerkung	Moderator*innen können nur Fragen in ihren eigenen Kursen anlegen. Es werden neue Daten in die Datenbank geschrieben, aber keine existierenden verändert oder gelöscht.

ID	9
Benutzerrolle	Admin
Name	Nutzerliste einsehen
Beschreibung	Als Admin möchte ich mir eine Übersicht über die vorhandenen Nutzer*innen anzeigen lassen, und für alle eine Detailansicht sehen können (die Implementierung dieser Ansicht erfolgt in US 10). Die hierfür benötigte Schlatfläche findet sich in einem, nur für Admins einsehbaren, Teil des Dashboards.
Akzeptanzkriterium	Eine Liste aller angemeldeten Nutzer*innen wird anch Klicken auf die entsprechende Schaltfläche angezeigt. Dies ist nur als Administrator*in möglich. Bei Klicken auf den Namen werden die Nutzer*innendetails angezeigt (US 10).
Story Points	3 Story Points
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	4
Tatsächlicher Aufwand	1:30 h
Velocity	0.5 h/SP
Bemerkung	Es werden keine Daten verändert, gelöscht oder erweitert. Eine Einsicht aller Nutzer*innen ist nur Administrator*innen möglich.

ID	10
Benutzerrolle	Admin
Name	Profil einsehen
Beschreibung	Als Admin der Seite möchte ich auf die Profile der Nutzer*innen Einsicht haben. Ich möchte aus einer Liste von Nutzer*innen eine*n Nutzer*in auswählen, anklicken und somit das betreffende Profil sehen können.
Akzeptanzkriterium	Administrator*innen können jedes Profil der Nutzer*innen einsehen, das Profil der Nutzer*innen wird korrekt dem Admin angezeigt.
Story Points	4 Story Points
Entwickler	Ilhan Simsiki
Umgesetzt in Iteration	4
Tatsächlicher Aufwand	10 h
Velocity	2.5 h/SP
Bemerkung	Einsicht auf die Profile fremder Nutzer*innen hat nur ein Admin. Es werden keine Daten geändert oder eingefügt.

ID	11
Benutzerrolle	Nutzer*in
Name	Moderationsrechte beantragen
Beschreibung	Als Nutzer*in möchte ich auf meiner Profilseite die Möglichkeit haben, Moderationsrechte zu beantragen. Ich klicke dazu auf dem Dashboard auf die Schaltfläche ”Moderationsrechte beantragen”, woraufhin mich die Webseite auffordert, eine Begründung einzugeben. Diese wird zusammen mit einem Link auf mein Profil an die Administartor*innen geschickt. Die Schaltfläche ist so lange nicht mehr auswählbar, bis ein*e Administrator*in mein Profil besucht und den Antrag angenommen oder abgelehnt hat.
Akzeptanzkriterium	Die Schaltfläche ist auswählbar, solange kein Antrag vorliegt. Beim Auswählen der Schaltfläche wird die Mail mit den oben genannten Informationen an die Administrator*innen verschickt. Liegt bereits innerhalb der letzten Woche ein Antrag vor, ist die Schaltfläche nicht auswählbar.
Story Points	8 Story Points
Entwickler	Claas Völcker
Umgesetzt in Iteration	8
Tatsächlicher Aufwand	5.5 h
Velocity	0.68 h/SP
Bemerkung	Es werden keine Daten geändert oder eingefügt.

ID	15
Benutzerrolle	Nutzer*in (bei Registrierung)
Name	Sprachauswahl
Beschreibung	Bei der Registrierung wählt der*die Nutzer*in auch eine Sprache aus, in der alle Inhalte der Webseite angezeigt werden sollen. Danach bekommt er nur Kurse seiner gewählten Sprache angezeigt.
Akzeptanzkriterium	Die Sprachauswahl wird mit dem*der Nutzer*in gespeichert und auf der Webseite berücksichtigt. Der*die Nutzer*in erhält als Erleichterung seiner Wahl eine Auskunft darüber, wie viele Kurse in der jeweiligen Sprache vorhanden sind.
Story Points	6
Entwickler	Ilhan Simsiki
Umgesetzt in Iteration	7
Tatsächlicher Aufwand	8 h
Velocity	1.14 h/SP
Bemerkung	Aktuell werden nur Deutsch und Englisch unterstützt, aber die Möglichkeit zum Anlegen weiterer Sprachen wird im Backend berücksichtigt (Stichwort: Erweiterbarkeit)

ID	14
Benutzerrolle	Nutzer*in
Name	Kurs fortführen
Beschreibung	Als Nutzer*in möchte ich einen bereits begonnenen Kurs fortführen können. Dazu gehe ich im Menü der Kursübersicht auf die Schalfläche ”Kurs fortführen”, um bei der ersten nicht beantworteten Frage fort zu fahren, oder wähle eine bereits beantwortete Frage aus, um an diesem Punkt mit der Kursbearbeitung fortzufahren.
Akzeptanzkriterium	Die Schaltfläche ”Kurs fortführen” wird angezeigt, sobald eine Frage in einem Kurs beantwortet wurde. Es ist nicht möglich Fragen in einem Kurs zu überspringen.
Story Points	8
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	7
Tatsächlicher Aufwand	7h
Velocity	0,88 h/SP
Bemerkung	Es werden keine Daten in der Datenbank geändert, hinzugefügt oder gelöscht.

ID	12
Benutzerrolle	Admin
Name	Moderationsrechte freischalten
Beschreibung	Als Admin möchte ich einer*einem beliebigen Nutzer*in meiner Wahl Moderatorenrechte gewähren, indem ich auf dem zugehörigen Profil die entsprechende Schaltfläche ”Moderationsrechte freigeben” auswähle.
Akzeptanzkriterium	Dem*der entsprechenden Nutzer*in werden Moderationsrechte beim Klicken auf die Schaltfläche gewährt. Wenn ein Moderationsantrag vorliegt und abgelehnt wird, kann der*die entsprechende Nutzer*in erneut Rechte beantragen.
Story Points	7 Story Points
Entwickler	Tobias Huber
Umgesetzt in Iteration	8
Tatsächlicher Aufwand	7,5 h
Velocity	
Bemerkung	Eventuelle Sicherheitsfeatures, wie bestätigung durch erneute Passworteingabe, werden in eine andere US ausgelagert. In der Datenbank werden Moderationsrechte gespeichert, aber keine anderen Daten verändert, erweitert oder gelöscht.

ID	19
Benutzerrolle	Nutzer*in
Name	Nutzerdetail ändern: E-Mail
Beschreibung	Als Nutzer*in möchte ich auf meinem Profil einstellen können, welche E-Mailadresse mit meinem Account verknüpft ist und dementsprechend zur Komunikation verwendet wird, indem ich im Abschnitt E-Mail das Textfeld neu ausfülle und auf ”Speichern” drücke.
Akzeptanzkriterium	Die Userstory wird akzeptiert, wenn der*die Nutzer*in seine E-Mailadresse ändern kann.
Story Points	3
Entwickler	Ilhan Siminski
Umgesetzt in Iteration	12
Tatsächlicher Aufwand	6,5 h
Velocity	0.46 h/SP
Bemerkung	Andere Daten werden nicht verändert.

ID	20
Benutzerrolle	Nutzer*in
Name	Nutzerdetail ändern: Passwort
Beschreibung	Als Nutzer*in möchte ich auf meinem Profil unter dem Abschnitt ”Passwort” mein eigenes Passwort neu festlegen können, indem ich das Textfeld neu ausfülle, mein neues Passwort noch einmal bestätige und auf ”Speichern” drücke. Zur Bestätigung dieses Vorgangs muss ich noch einmal mein aktuelles Passwort eingeben.
Akzeptanzkriterium	Die Userstory wird akzeptiert, wenn der*die Nutzer*in sein*ihr Passwort ändern und sich mit diesem dann auf der Website einloggen kann.
Story Points	4
Entwickler	Tobias Huber
Umgesetzt in Iteration	12
Tatsächlicher Aufwand	6 h
Velocity	1.5 h/SP
Bemerkung	Die Wiederherstellung eines vergessenen Passworts wird in einer anderen Userstory behandelt.

ID	4
Benutzerrolle	Moderator*in
Name	Kurs bearbeiten
Beschreibung	Als Moderator*in möchte ich meine eigenen Kurse bearbeiten können. Dafür soll in der Kursübersicht bei meinen Kursen ein Feld "Kurs bearbeiten" zum Anklicken sein. Nach Anklicken erscheint ein Bearbeitungsfenster, auf dem ich Titel, Zuordnung des Kurses und die einzelnen Module des Kurses bearbeiten kann. Wenn ich die Bearbeitung ohne zu speichern abbreche, wird der Kurs wieder auf seinem ursprünglichen Zustand gesetzt.
Akzeptanzkriterium	Der Editor ist für Moderator*inen des Kurses und Administrator*innen des Kurses nutzbar. Er ist funktional identisch zum "Kurs anlegen"-Editor. Der bearbeitete Kurs wird korrekt in der Datenbank abgespeichert und auf der Seite aktualisiert. Moderator*inen können nur ihre eigenen Kurs bearbeiten. Wird der Änderungsprozess abgebrochen, ändern sich die Daten in der Datenbank nicht.
Story Points	8
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	12
Tatsächlicher Aufwand	8,25 h
Velocity	1.03 h/SP
Bemerkung	Der bearbeitete Kurs wird in der Datenbank geändert. Dadurch werden auch abhängige Daten (Module und Frage geändert).

ID	17
Benutzerrolle	Nutzer*in
Name	Nutzerdetail ändern: Sprache
Beschreibung	Als Nutzer*in möchte ich auf meinem Profil einstellen können, ob ich Englisch oder Deutsch als allgemeine Sprache der Website bevorzuge, indem ich vorher einen Reiter "Einstellungen" öffne und dort im Abschnitt "Sprache" den Knopf mit der entsprechenden Sprache auswähle.
Akzeptanzkriterium	Die Userstory wird akzeptiert, wenn der*die Nutzer*in nach erfolgreicher Änderung die Website auf der korrekten Sprache angezeigt bekommt, sofern Übersetzungen vorliegen. Wird der Vorgang abgebrochen, werden keine Daten geändert.
Story Points	3
Entwickler	Ilhan Siminski
Umgesetzt in Iteration	10
Tatsächlicher Aufwand	5 h
Velocity	1.66 h/SP
Bemerkung	Andere Daten bleiben von den Änderungen unberührt.

ID	18
Benutzerrolle	Nutzer*in
Name	Nutzerdetail ändern: Name
Beschreibung	Als Nutzer*in möchte ich auf meinem Profil sowohl meinen eigenen (Vor- und Nach-) Namen, als auch meinen Username ändern können, indem ich vorher einen Reiter "Einstellungen" öffne und dort im Abschnitt "Namen" die Textfelder neu ausfülle und auf Speichern" drücke.
Akzeptanzkriterium	Die Userstory wird akzeptiert, wenn der Name in der Datenbank nach Eingeben des korrekten Passworts persistent gespeichert wird. Wird der Vorgang abgebrochen, wird nichts geändert.
Story Points	3
Entwickler	Tobias Huber
Umgesetzt in Iteration	10
Tatsächlicher Aufwand	4 h
Velocity	1.33 h/SP
Bemerkung	Der Username kann nicht geändert werden, da er als eindeutige Authentifikation gegenüber anderen Nutzer*innen der Plattform dient. Die erfolgreiche Bestätigung ändert den Namen des*der Nutzer*in im Profil in der Datenbank. Andere Daten bleiben unberührt.

ID	22
Benutzerrolle	Moderator*in
Name	Multiple Choice Bilder einpflegen
Beschreibung	Als Moderator*in möchte ich bei einer Multiple Choice Frage zur Frage und zum Feedback ein Bild anzeigen lassen. Dazu möchte ich beim Fragen Erstellen die Möglichkeit haben, ein Bild für die Frage und eines für das Feedback hochzuladen.
Akzeptanzkriterium	Zum Hochladen der Bilder wird jeweils ein Popup angezeigt, in dem eine Schaltfläche angezeigt wird. Diese öffnet ein Fenster, in dem die Datei vom lokalen Rechner ausgewählt werden kann. Nach dem Hochladen wird eine Vorschau des Bildes angezeigt, auf der der gewünschte Bildausschnitt ausgewählt werden kann.
Story Points	5
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	11
Tatsächlicher Aufwand	4,5
Velocity	1,1
Bemerkung	Die Daten werden beim Speichern des zugehörigen Kurses persistent in der Datenbank geändert.

ID	16
Benutzerrolle	Nutzer*in
Name	Kursfortschritt anzeigen (Übersicht)
Beschreibung	Als Nutzer*in möchte ich meinen Fortschritt in einem Kurs in der Kursübersicht einsehen können.
Akzeptanzkriterium	Die Schaltfläche zur Kurswahl wird prozentual zum Fortschritt eines*einer Nutzer*in im jeweiligen Kurs eingefärbt. In der Kursübersicht wird die Anzahl der beantworteten Fragen und die Gesamtzahl an Fragen im Kurs angezeigt.
Story Points	5
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	7
Tatsächlicher Aufwand	5 h
Velocity	0.71 h/SP
Bemerkung	Es werden keine Daten geändert, hinzugefügt oder gelöscht.

ID	23
Benutzerrolle	Moderator*in
Name	Einklappen von Modul Komponenten ermöglichen
Beschreibung	In der Kursbearbeiten-Übersicht soll jede Komponente Module ein und ausklappbar sein. Dazu soll neben der Überschrift der jeweiligen Komponente eine Schaltfläche vorhanden sein. Ein Klick auf diese ermöglicht, abhängig vom aktuellen Zustand, das Ein- und Ausklappen.
Akzeptanzkriterium	Die Schaltfläche ist für jede Komponete vorhanden und durch das Klicken wird nur die betreffende Komponente animiert.
Story Points	8
Entwickler	Ilhan Siminski
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	8
Velocity	1 h/SP
Bemerkung	Es werden keine Daten geändert, neu hinzugefügt oder gelöscht. Die Fragen übersichtlicher zu machen wurde in US 58 ausgelagert.

ID	24
Benutzerrolle	Moderator*in
Name	Info Fragetyp einpflegen
Beschreibung	Als Moderator möchte ich einen Informationstext anstelle einer Frage einfügen können. Dieser soll im Editor wie eine normale Frage anlegbar und bearbeitbar sein. Anstelle von möglichen Antworten kann ein längerer Text eingefügt werden.
Akzeptanzkriterium	Der Info-Typ kann im Editor wie eine Frage hinzugefügt und bearbeitet werden und wird nach Speichern persistent in der Datenbank angelegt.
Story Points	3
Entwickler	Claas Völcker
Umgesetzt in Iteration	13
Tatsächlicher Aufwand	2,5 h
Velocity	0.83 h/SP
Bemerkung	Nach Speichern des Kurses wird der Informationstext persistent in der Datenbank gespeichert.

ID	25
Benutzerrolle	Nutzer*in
Name	Info Fragetyp anzeigen
Beschreibung	Als Nutzer*in möchte ich beim Bearbeiten des Kurses Fragen vom Typ Information angezeigt bekommen.
Akzeptanzkriterium	Dem*der Nutzer*in werden Fragen vom Typ Information angezeigt und er kann sie bearbeiten, indem er auf die Schlatfläche "Fortfahren" klickt. Die zu jeder Frage gehörenden Informationen werden im linken Drittel des Fensters und der Text der Frage auf den rechten zwei Dritteln angezeigt.
Story Points	5
Entwickler	Claas Völcker
Umgesetzt in Iteration	14
Tatsächlicher Aufwand	5,5 h
Velocity	1.1 h/SP
Bemerkung	Durch Auswählen der Schlatfläche ”Fortfahren” wird die Frage in der Statistik des*der Nutzer*in gespeichert.

ID	26
Benutzerrolle	Nutzer*in
Name	Informationsseite
Beschreibung	Als Nutzer*in möchte ich Informationen über die Webseite und das zugrunde liegende iGEM Projekt angezeigt bekommen, wenn ich im entsprechenden Reiter auf die Schaltfläche "About" klicke.
Akzeptanzkriterium	Die Informationsseite kann als HTML hinterlegt werden und wird korrekt dargestellt.
Story Points	2
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	13
Tatsächlicher Aufwand	2 h
Velocity	1 h / SP
Bemerkung	Die Bereitstellung der HTML Seiten ist nicht Teil des Projektes. Diese können von den Auftraggeber*innen nach Absprache slebst zur Verfügung gestellt werden, indem die betreffenden Dateien auf dem Server im dafür vorgesehenen Ordner hinterlegt werden.

ID	28
Benutzerrolle	Admin
Name	Statistik exportieren
Beschreibung	Als Admin habe ich die Möglichkeit, die gesammte Nutzungsstatistik der Seite zu exportieren.
Akzeptanzkriterium	Es gibt eine Schaltfläche in der Statistik Übersicht, über welche ich mir alle geloggten Einträge der Statistik als *.csv exportieren kann.
Story Points	4
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	13
Tatsächlicher Aufwand	3,75h
Velocity	0,94 h/SP
Bemerkung	Die Userstory Anonyme Statistik ist hieraus entstanden.

ID	27
Benutzerrolle	Nutzer*in
Name	Kurs Übersicht anzeigen
Beschreibung	Als Nutzer*in kann ich mir eine Kursübersicht anzeigen lassen. Die dafür vorgesehene Schaltfläche findet sich auf der Übersichtsseite. Diese enthält alle Fragen des Kurses, den aktuellen Fortschritt und Schaltflächen um richtig bearbeitete Fragen zu wiederholen.
Akzeptanzkriterium	Es gibt eine Schaltfläche in jeder Frage um das Dashboard zu öffnen. Das Dashboard enthällt alle Fragen und den aktuellen Fortschritt.
Story Points	6
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	11
Tatsächlicher Aufwand	6,5h
Velocity	1,08 h/SP
Bemerkung	

ID	28
Benutzerrolle	Admin
Name	Statistik exportieren
Beschreibung	Als Admin habe ich die Möglichkeit, die gesammte Nutzungsstatistik der Seite zu exportieren.
Akzeptanzkriterium	Es gibt eine Schaltfläche in der Statistik Übersicht, über welche ich mir alle geloggten Einträge der Statistik als *.csv exportieren kann.
Story Points	4
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	13
Tatsächlicher Aufwand	3,75h
Velocity	0,94 h/SP
Bemerkung	Die Userstory Anonyme Statistik ist hieraus entstanden.

ID	29
Benutzerrolle	Nutzer*in
Name	Passwort anfordern
Beschreibung	Als Nutzer*in der Platform habe ich die Möglichkeit, ein neues Passwort anzufordern, falls ich mein altes vergessen habe. Dafür soll es im Login Bereich einen Button geben, mit dem sich ein Fenster öffnet. Dort gebe ich die E-Mail Adresse an, die mit meinem Account verlinkt ist, und erhalte eine E-Mail mit meinem Nutzernamen und einem neu generierten Passwort.
Akzeptanzkriterium	Wenn der*die Nutzer*in seine E-Mail Adresse angegeben hat, welche auch in der Datenbank hinterlegt ist wird eine E-Mail mit dem neu generierten Passwort und dem Profilnamen an die E-Mail gesendet. Ansonsten wird eine aussagekräftige Fehlermeldung angezeigt.
Story Points	6
Entwickler	Tobias Huber
Umgesetzt in Iteration	19
Tatsächlicher Aufwand	8h
Velocity	ca. 1.33h/SP
Bemerkung	Es werden keine Felder in der Datenbank geändert.

ID	30
Benutzerrolle	Admin
Name	Moderatorenrechte entziehen
Beschreibung	Als Admin möchte ich einem Moderator seine Moderatorenrechte wieder entziehen können. Der Moderator soll dann zu einem normalen Nutzer werden.
Akzeptanzkriterium	Im Profil des*der Moderator*in ist für Administartor*innen eine Schaltfläche vorhanden, die es ermöglicht, die Moderationsrechte zu entziehen. Nach Klicken auf diese Schalfläche wird de*die Nutzer*in aus der Gruppe "Moderatoren" gelöscht und verliert Zugriff auf alle Menus, die nur für Moderator*innen einsehbar sind. Die Kurse, die diese*r Moderator*in bislang betreut hat, sind vorläufig nur noch von Administartor*innen änderbar.
Story Points	6
Entwickler	Tobias Huber
Umgesetzt in Iteration	16
Tatsächlicher Aufwand	7h
Velocity	1.16 h/SP
Bemerkung	Beim Klicken auf die Schlatfläche wird in der Datnbank die Verknüpfung zwischen Gruppe "Moderator" und dem Nutzer gelöscht.

ID	31
Benutzerrolle	Admin
Name	Adminrechte Vergeben
Beschreibung	Als Admin möchte ich einem*einer Nutzer*in oder Moderator*in zum Admin machen können. Dafür gibt es in der Admin-Ansicht in der Übersicht der Nutzer*in einen Button um eine*n Moderator*in oder Nutzer*in zum Admin zu befördern.
Akzeptanzkriterium	Nach dem Befördern eines*einer Nutzer*in (oder Moderator*in) wird dieser in die Gruppe "Admin" aufgenommen. Er*sie kann danach alle Aktionen ausführen, die ein Admin ausführen darf.
Story Points	6
Entwickler	Tobias Huber
Umgesetzt in Iteration	17
Tatsächlicher Aufwand	7 h
Velocity	1.16 h/SP
Bemerkung	In der Datenbank wird der*die beförderte Nutzer*in der Gruppe "Administrator" hinzugefügt.

ID	32
Benutzerrolle	Admin
Name	Adminrechte entziehen
Beschreibung	Als Admin möchte ich einem Moderator*in die Adminrechte wieder entziehen können. Wenn ein*e Admin eine*n andere*n Admin in der Adminansicht auswählt, gibt es dort einen Button, mit dem man dem*der Admin seine*ihre Adminrechte entziehen kann.
Akzeptanzkriterium	Nach Entzug der Adminrechte kann der*die betreffende Nutzer*in keine administrativen Aufgaben mehr erledigen. Er*sie verliehrt Zugriff auf alle Bereiche der Seite und Schaltflächen, die nur für Administrator*innen sichtbar sind. Wenn er*sie davor Moderator*in war, behält er*sie die Rechte dieser Gruppe weiterhin.
Story Points	3
Entwickler	Tobias Huber
Umgesetzt in Iteration	16
Tatsächlicher Aufwand	4h
Velocity	
Bemerkung	Beim Entzug der Adminrechte wird die Verknüpfung zwischen dem*der betreffenden Nutzer*in und der Gruppe "Admin" in der Datenbank gelöscht.

ID	33
Benutzerrolle	Admin
Name	Kategorien bearbeiten
Beschreibung	Als Admin möchte ich die Möglichkeit haben, existierende Kategorien zu bearbeiten. Dafür gibt es im Admin-Bereich den Menüpunkt "Kategorien bearbeiten". In diesem werden die Kategorien aufgelistet. Ich kann hier existierende Kategorien bearbeiten.
Akzeptanzkriterium	Wenn eine Kategorie bearbeitet wird, werden die Werte in der Datenbank aktualisiert. Bei Bearbeiten der Kategorie werden alle Kurse, welche in der betreffenden Kategorie sind, dem neuen Kategorienamen zugeordnet.
Story Points	2
Entwickler	Tobias Huber
Umgesetzt in Iteration	17
Tatsächlicher Aufwand	4 h
Velocity	2 h/SP
Bemerkung	Wird der Prozess abgebrochen, so werden keine Daten in der Datenbank geändert. Nur die gewünschte Kategorie wird geändert, alle anderen Daten bleiben unbetroffen.

ID	34
Benutzerrolle	Admin
Name	Neue Kategorie anlegen
Beschreibung	Als Admin möchte ich Kategorien anlegen können, die die Webseite benötigt. Dazu gibt es eine Schaltfläche im Bereich ”Kategorien bearbeiten”.
Akzeptanzkriterium	Beim Klicken auf die Schlatfläche gibt es ein Fenster, in dem der Name der neuen Kategorie und eine zugeordnete Farbe eingetragen werden können. Nach erneutem Klicken wird die Kategorie in der Datenbank gespeichert und an allen entsprechenden Stellen auf der Webseite angezeigt.
Story Points	3
Entwickler	Tobias Huber
Umgesetzt in Iteration	17
Tatsächlicher Aufwand	4 h
Velocity	1.33 h/SP
Bemerkung	

ID	35
Benutzerrolle	Admin
Name	Kategorie löschen
Beschreibung	Als Administator möchte ich Kategorien löschen können, die die Webseite nicht mehr benötigt. Dazu gibt es eine Schaltfläche im Bereich "Kategorien bearbeiten".
Akzeptanzkriterium	Im Adminbereich "Kategorien bearbeiten" gibt es eine Schaltfläche zum Löschen von Kursen. Bei Löschung einer Kategorie werden auch alle Kurse dieser Kategorie gelöscht. Um ein versehentliches Löschen von Kursen zu verhindern, wird beim Klick auf die Schaltfläche ein Warnfenster angezeigt, in dem alle Kurse aufgelistet sind, die gelöscht würden. Der Administartor muss den Vorgang dann noch einmal bestätigen, bevor die Kategorie tatsächlich gelöscht wird. Wenn keine Kategorien existieren soll im Dashboard kein Kurs angezeigt werden.
Story Points	4
Entwickler	
Umgesetzt in Iteration	
Tatsächlicher Aufwand	
Velocity	
Bemerkung	Wurde im Rahmen des Projektes nicht abgegeben.

ID	36
Benutzerrolle	Nutzer*in
Name	Footer auf der Hauptseite
Beschreibung	Auf der Hauptseite wird ein Footer mit generellen Infos über die Seite angezeigt. Darin werden Links auf die Social Media Plattformen der Betreiber*innen, ein Link auf das Github Projekt der Seite und Links auf die statischen Seiten der Webseite (aktuell: ”About”, ”Impressum”, ”Contact”) bereitgestellt.
Akzeptanzkriterium	Der Footer wird auf allen Seiten unten auf der Seite angezeigt (auf einigen werden allerdings die Links ersetzt (US 37). Die Links sind anklickbar und zeigen auf die richtigen URLs.
Story Points	3
Entwickler	Leonhardt Wiedmann
Umgesetzt in Iteration	16
Tatsächlicher Aufwand	3,5
Velocity	1.16 h/SP
Bemerkung	Es werden keine Daten in der Datenbank, geändert, hinzugefügt oder gelöscht.

ID	37
Benutzerrolle	Nutzer*in
Name	Footer auf der Frageseite
Beschreibung	Um die Navigation auf der Frageseite zu vereinfachen, wird auf der Frageseite ein Footer angezeigt, in dem der aktuelle Fortschritt im Kurs in Form eines Balkens ersichtlich ist. Dieser ist passend zum aktuellen Fortschritt des Nutzers prozentual eingefärbt. Zusätzlich wird links eine Schaltfläche angezeigt, die das Sidenav zur Navigation im Kurs öffnet (US 42). Rechts wird die Schaltfläche zum Abgeben der aktuellen Fragen bzw. zur Navigation zur nächsten angezeigt.
Akzeptanzkriterium	Der Footer wird dauerhaft unten auf der Frageseite angezeigt. Alle Schaltflächen werden angezeigt und zeigen das oben beschriebene Verhalten. Der Fortschrittsbalken ist zu jedem Zeitpunkt entsprechend des aktuellen Fortschritts gefüllt.
Story Points	5
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	16
Tatsächlicher Aufwand	5
Velocity	1 h/SP
Bemerkung	Es werden keine Daten in der Datenbank, geändert, hinzugefügt oder gelöscht.

ID	38
Benutzerrolle	Nutzer*in
Name	Dashboard Ansicht strukturieren
Beschreibung	Wenn ein Nutzer*in auf die Seite /courses/ kommt, soll für ihn*sie ersichtlich sein, welche Kurse er*sie zuletzt bearbeitet und noch nicht abgeschlossen hat.
Akzeptanzkriterium	Sobald ein*e Nutzer*in die Seite /courses/ aufruft, werden ihm alle von ihm angefangenen Kurse angezeigt. Die Übersicht enthält für jeden Kurs die folgenden Informationen: Kurstitel, aktuelles Modul, Prozentsatz des Kurses, welcher bereits abgeschlossen wurde. Durch Auswählen einer Schaltfläche kann der*die Nutzer*in sofort mit der Bearbeitung des Kurses fortfahren.
Story Points	8
Entwickler	Claas Völcker
Umgesetzt in Iteration	15
Tatsächlicher Aufwand	9 h
Velocity	1.125 h/SP
Bemerkung	Es werden keine Daten in der Datenbank, geändert, hinzugefügt oder gelöscht.

ID	39
Benutzerrolle	Moderator*in
Name	Dashboard Menu für Moderator*innen
Beschreibung	Um die Seite für Moderator*innen übersichtlicher zu gestalten, soll im Dashboard ein Dropdown Menu erscheinen, in dem die von diesem*dieser Moderator*in bearbeiteten Kurse angezeigt werden.
Akzeptanzkriterium	Im Menü im Dashboard erscheint ein extra Reiter, wenn man als Moderator*in eingelogt ist. Dort werden alle Kurse aufgelistet, die der*die Moderator*in erstellt hat. Dies können wie normale Kurse zum Bearbeiten oder Ändern ausgewählt werden.
Story Points	5
Entwickler	Ilhan Simsiki
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	7,5h
Velocity	1.5h/SP
Bemerkung	Für die Übersicht werden keine Daten in der Datenbank verändert, hinzugefügt oder gelöscht.

ID	40
Benutzerrolle	Nutzer*in
Name	InfoText mit Youtube Video anzeigen
Beschreibung	Dem*der Nutzer*in soll ein Informationstext anstelle einer Frage angezeigt werden, bei dem ein YouTube Video 2/3 Drittel der Seite einnimmt. Beim Klicken auf dieses Video wird es abgespielt.
Akzeptanzkriterium	Wenn der*die Nutzer*in eine Informationstext-Frage anzeigt, die ein YouTube Video enthält, wird das Video auf den rechten 2/3 der Seite angezeigt. Links wird der reguläre Informationstext angezeigt (siehe US 25). Beim Klicken auf die Schaltfläche "Weiter" wird die Frage als Beantwortet in die Statistik eingetragen und der*die Nutzer*in kann mit der Bearbeitung des Kurses fortfahren.
Story Points	4
Entwickler	Claas Völcker
Umgesetzt in Iteration	16
Tatsächlicher Aufwand	4 1/2 h
Velocity	1.25 h/SP
Bemerkung	Bei Klicken auf die Schaltfläche "Weiter" wird persistent in der Datenbank gespeichert, dass die Frage richtig beantwortet wurde.

ID	41
Benutzerrolle	Moderator*in
Name	InfoText mit Youtube Video einpflegen
Beschreibung	Als Moderator*in möchte ich die Möglichkeit haben, einen Informationstext mit einem YouTube Video anstelle einer Frage anzulegen. Im Menu "Kurs anlegen/bearbeiten" kann ich dazu als Frage "InfoText (YouTube)" auswählen und einen Informationstext hinterlegen (siehe US 24). Zusätzlich erhalte ich die Möglichkeit einen Link einzutragen, der auf ein YouTube Video verweißt. Dieser wird in der Datenbank gespeichert.
Akzeptanzkriterium	Beim Anlegen des Informationstextes wird in Feld angezeigt, in welches ein YouTube Link geschrieben werden kann. Dieser wird beim Speichern darauf überprüft, ob es tatsächlich ein gültiger Video-Link ist und dann gespeichert. Ist es kein gültiger Link, wird eine entsprechende Fehlermeldung angezeigt.
Story Points	4
Entwickler	Claas Völcker
Umgesetzt in Iteration	16
Tatsächlicher Aufwand	3 1/2 h
Velocity	0.875 h/SP
Bemerkung	Nach Speichern des Kurses wird der Informationstext persistent in der Datenbank gespeichert.

ID	42
Benutzerrolle	Nutzer*in
Name	Sidenav Frageseite erstellen
Beschreibung	Für jede*n Nutzer*in soll in der Frageseite das Sidenav-Menü aufrufbar sein. Dieses Menü soll so aussehen, wie es im Protokoll vom 20.08 definiert wurde.
Akzeptanzkriterium	Wenn man in der Ansicht der Fragen im Footer auf den Button für das Dashboard klickt, öffnet dieser sich auf der linken Seite des Bildschirms mit den Inhalten, wie im Protokoll definiert.
Story Points	5
Entwickler	Claas Völker
Umgesetzt in Iteration	17
Tatsächlicher Aufwand	5,5
Velocity	1.11 h/SP
Bemerkung	Es werden keine Daten in der Datenbank, geändert, hinzugefügt oder gelöscht.

ID	43
Benutzerrolle	Moderator*in
Name	Quiz anlegen
Beschreibung	Zum Abschluss eines Kurses möchte ein*e Moderator*in ein Quiz zu seinem Kurs hinzufügen.
Akzeptanzkriterium	Als Moderator*in kann man beim Bearbeiten seines Kurses am Ende ein Quiz hinzufügen. Dieses Quiz besteht aus 5 - 20 Fragen. Jede Frage beinhaltet eine Fragentext, 4 verschiedene Antwortmöglichkeiten und optional ein Bild. Wenn der*die Moderator*in den Kurs mit Quiz speichert, wird dieser persistent in der Datenbank hinterlegt.
Story Points	7
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	17
Tatsächlicher Aufwand	6 h
Velocity	0.857h/SP
Bemerkung	Nach Speichern des Kurses wird das Quiz und alle Fragen persistent in der Datenbank gespeichert.

ID	44
Benutzerrolle	Nutzer*in
Name	Quiz anzeigen & beantworten
Beschreibung	Um einen Kurs komplett abzuschließen, soll am Ende des Kurses ein Quiz erscheinen. Dieses unterscheidet sich optisch von den regulären Fragen und es soll nicht möglich sein, während dem Quiz im Kurs zu navigieren.
Akzeptanzkriterium	Nachdem ein*e Nutzer*in alle Module eines Kurses abgeschlossen hat, wird diesem*dieser das Quiz für den Kurs angezeigt. Dieses besteht aus Frage, Antwortmöglichkeit und optional einem Bild. Wenn der*die Nutzer*in das Quiz abgeschlossen hat, wird angezeigt wie gut diese*r abgeschnitten hat. Zusätzlich werden Statistik und Ranking in der Datenbank abgespeichert.
Story Points	5
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	17
Tatsächlicher Aufwand	4,25 h
Velocity	0.85 h/SP
Bemerkung	Beim Absenden des Quizes wird die Statistik des*der Nutzer*in persistent in der Datenbank gespeichert.

ID	45
Benutzerrolle	Nutzer*in
Name	Eigenes Ranking anzeigen
Beschreibung	Nutzer*innen haben die Möglichkeit, ihre bisherige, durch Beantwortung von Fragen erreichte Punktzahl einzusehen. Alle Fragen, die eine Beantwortung erfordern (Multiple Choice & Quiz) geben dem*der Nutzer*in Punkte. Diese werden im Profil gespeichert und auf der eigenen Profilseite angezeigt.
Akzeptanzkriterium	Bei der Beantwortung der oben genannten Fragetypen erhalten die Nutzer*innen Punkte, die in ihrem Profil gespeichert werden. Die Punktzahl wird in der Profilansicht angezeigt.
Story Points	3
Entwickler	Ilhan Simsiki
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	5h
Velocity	1.25 h/SP
Bemerkung	Die Daten über das Ranking werden bei Beantwortung der Fragen persistent in der Profiltabelle der Datenbank gespeichert.

ID	46
Benutzerrolle	Nutzer*in
Name	Globales Ranking anzeigen
Beschreibung	Nutzer*innen können das gesamte Ranking auf einer eigenen Seite einsehen. Auf dieser werden alle Nutzer*innen anhand ihrer Punktzahl im Ranking (siehe US 45) sortiert.
Akzeptanzkriterium	Auf der Profilseite und in der Navigationleiste gibt es einen Link, der auf die Übersichtsseite des Rankings führt. Dort werden alle Nutzer*innen mit ihrer Punktzahl sortiert aufgelistet. Nutzer*innen, die nicht in der Statistik auftauchen möchten, können dies in ihren Einstellungen ändern. Sie verlieren dadurch auch den Zugriff auf die Übersichtsseite.
Story Points	7
Entwickler	Claas Völcker
Umgesetzt in Iteration	17
Tatsächlicher Aufwand	6,5 h
Velocity	0.928 h/SP
Bemerkung	Für die Berechnung des Rankings werden keine Daten in der Datenbank geändert.

ID	47
Benutzerrolle	Nutzer
Name	Wöchentliche Statistik für Nutzer
Beschreibung	Ein Nutzer möchte seine Statistik sehen. Dafür kann gibt es eine Seite in der die Statistik der letzten 7 Tage und wie viele Fragen er richtig beantwortet hat.
Akzeptanzkriterium	
Story Points	7
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	7,25 h
Velocity	
Bemerkung	

ID	48
Benutzerrolle	Moderator*in
Name	Statistiken für einen Kurs exprotieren
Beschreibung	Um eine Übersicht über ihre Kurse zu erhalten, möchten Moderator*innen eine Statistik exportieren können. Diese Statistik soll anzeigen, wie viele Nutzer*innen den Kurs bearbeitet und wie gut diese abgeschnitten haben.
Akzeptanzkriterium	Wenn man als Moderator*in einen Kurs im Dashboard auswählt, sieht man einen zusätzlichen Button für die Statistik. Dieser führt zu einer Seite, welche die Statistik des Kurses anzeigt. Es werden die einzelnen Fragen, Module und die Statistik des gesamten Kurses angezeigt.
Story Points	7
Entwickler	Leonhard Wiedmann
Umgesetzt in Iteration	19
Tatsächlicher Aufwand	6.5h
Velocity	0.93 h/SP
Bemerkung	Es werden keine Daten in der Datenbank, geändert, hinzugefügt oder gelöscht.

ID	49
Benutzerrolle	Nutzer*in
Name	Profile Quickview im Header
Beschreibung	Ein*e Benutzer*in der Seite möchte schnell einen Überblick über sein*ihr Profil erhalten oder sich schnell ausloggen.
Akzeptanzkriterium	Ein*e Benutzer*in kann über das Menü ein kleines Fenster öffnen in dem ein Überblick über seine*ihre Statistik und Ranking angezeigt wird, so wie Buttons zum Ausloggen und um zu den Einstellungen zu gelangen.
Story Points	5
Entwickler	Ilhan Simsiki
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	8h
Velocity	1.6 h/SP
Bemerkung	Es werden keine Daten in der Datenbank, geändert, hinzugefügt oder gelöscht.

ID	50
Benutzerrolle	Admin
Name	Anonymisierung der Statistic
Beschreibung	Beim Anzeigen der Statistik für Moderator*innen und Administrator*innen werden Benutzer*innennamen nur anonymisiert gezeigt. Dadurch wird der Schutz der persönlichen Daten gegenüber den Betreiber*innen der Seite gewährleistet. Das Pseudonym wird für jedes Profil angelegt und für alle Statistiken verwendet, damit man Kontinuitäten erkennen kann.
Akzeptanzkriterium	Jede*r Nutzer*in erhält bei Registrierung ein Pseudonym, welches bei Statistiken angezeigt wird, die sich Moderator*innen und Administrator*innen anzeigen lassen können.
Story Points	5
Entwickler	Claas Völcker
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	5,75 h
Velocity	1.15 h/SP
Bemerkung	Das Pseudonym wird aus konstanten Daten der Datenbank errechnet (Webseitenschlüssel und Nutzer*innenname), aber selbst nicht gespeichert.

ID	
Benutzerrolle	Nutzer*in
Name	User Bild hochladen
Beschreibung	Ein*e Nutzer*in möchte für das Profil ein eigenes Bild hochladen. Dazu kann er*sie in den Einstellungen ein Profilbild hochladen.
Akzeptanzkriterium	Als Benuzter*in kann ich im Bereich "Einstellungen" ein Bild hochladen, welches in der Datenbank gespeichert wird.
Story Points	6
Entwickler	Ilhan Simsiki
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	11h
Velocity	1.83 h/SP
Bemerkung	Das Bild wird persistent im Profil des Nutzers gespeichert. Andere Daten sind davon nicht betroffen.

ID	52
Benutzerrolle	Nutzer*in
Name	Quiz Feedback anzeigen
Beschreibung	Nach einer Quizfrage wird dem*der Benutzer*in das Ergebnis angezeigt.
Akzeptanzkriterium	Wenn ein*e Nutzer*in eine Quizfrage abgeschlossen hat, wird ihm*ihr das Ergebnis seiner*ihrer Antwort gezeigt. Er*sie kann sehen, welche Antwort richtig gewesen wäre und dann die nächste Frage bearbeiten. Er*sie hat keine Möglichkeit, die Frage direkt zu wiederholen.
Story Points	5
Entwickler	Claas Voelcker
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	5,5 h
Velocity	1.1 h/SP
Bemerkung	Es werden keine Daten in der Datenbank, geändert, hinzugefügt oder gelöscht.

ID	53
Benutzerrolle	Admin
Name	Kurs löschen
Beschreibung	Admins haben die Möglichkeit Kurse zu löschen. Bei der Übersicht des Kurses gibt es dafür eine Schaltfläche. Beim Klicken auf diese wird der Kurs mit allen Modulen und Fragen gelöscht.
Akzeptanzkriterium	Um das versehentliche Löschen eines Kurses zu verhindern, wird vor dem endgültigen Löschen ein Popup mit der Bitte um Bestätigung angezeigt. Erst nach erneuter Bestätigung werden Kurs und alle abhängigen Daten (Module, Fragen und Quiz) gelöscht. Statistiken über die Fragen bleiben jedoch erhalten, damit der Nutzer seine Erfolge weiterhin sehen kann. Wird der Prozess in einem Schritt vor der endgültigen Bestätigung durch den Adiminstrator unterbrochen, werden keine Daten geändert.
Story Points	3
Entwickler	Tobias Huber
Umgesetzt in Iteration	
Tatsächlicher Aufwand	
Velocity	
Bemerkung	Wurde im Rahmen des Projektes nicht mehr umgesetzt.

ID	54
Benutzerrolle	Admin
Name	Kurs sichtbar schalten
Beschreibung	Ein*e Admin kann einen Kurs sichtbar schalten. Dadurch wird er für alle Nutzer im Dashboard angezeigt.
Akzeptanzkriterium	Wenn ein Kurs von unsichtbar zu sichtbar geschalten wird, wird er richtig für alle Nuzter*innen im Dashboard angezeigt. Ein unsichtbarer Kurs wird nur für Moderator*innen oder Admins angezeigt.
Story Points	3
Entwickler	Tobias Huber
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	4h
Velocity	1.25 h/SP
Bemerkung	Das Feld ”Sichtbarkeit” des KUrses wird in der Datenbank geändert. Andere daten sind davon nicht betroffen.

ID	
Benutzerrolle	Moderator*in
Name	Kurs invisible schalten
Beschreibung	Ein*e Moderator*in möchte bereits angelegte Kurse wieder "invisible" schalten dürfen, also so umschalten, dass der Kurs zwar nicht gelöscht, jedoch auch nicht mehr für Nutzer*innen angezeigt wird.
Akzeptanzkriterium	Wenn ich als Moderator*in in der Kursübersicht auf einen Knopf drücken, der mit einem geöffneten Auge versehen ist, wird daraufhin stadtdessen ein geschlossenes Auge angezeigt und der Kurs wird für normale Nutzer*innen unzugänglich.
Story Points	3
Entwickler	Tobias Huber
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	4h
Velocity	1.25 h/SP
Bemerkung	Das Feld ”Sichtbarkeit” des KUrses wird in der Datenbank geändert. Andere daten sind davon nicht betroffen.

ID	56
Benutzerrolle	Nutzer*innen
Name	Punkte des Rankings im Quiz berechnen
Beschreibung	Quizfragen geben zusätzliche Punkte im Ranking über die normalen Beantwortungspunkte hinaus. Pro 5 richtig beantworteter Fragen gibt es zusätzliche Extrapunkte.
Akzeptanzkriterium	Die Punkte der Quizfragen werden nach dem verabredeten Schema (siehe Dokument von Timo) berechnet.
Story Points	4
Entwickler	Claas Voelcker
Umgesetzt in Iteration	20
Tatsächlicher Aufwand	4
Velocity	1 h/SP
Bemerkung	Das Feld ”Sichtbarkeit” des KUrses wird in der Datenbank geändert. Andere daten sind davon nicht betroffen.

ID	56
Benutzerrolle	Nutzer*innen
Name	Zweite Authentifizierung bei Profiländerungen
Beschreibung	Wenn ein*e Nutzer*in eine Änderung an seinem*ihrem Profil vornehmen möchte, muss er*sie sich erneut authentifizieren.
Akzeptanzkriterium	Die Userstory wird akzeptiert, wenn sich beim Klicken ein Popup öffnet
Story Points	5
Entwickler	Tobias Huber
Umgesetzt in Iteration	19
Tatsächlicher Aufwand	5
Velocity	1 h/SP
Bemerkung	Es werden durch diese Userstory keine Daten in der Datenbank, geändert, hinzugefügt oder gelöscht. Die Änderungen sind in den betreffenden Userstories zu Profileinstellungen spezifiziert.

ID	58
Benutzerrolle	Moderator*in
Name	Einklappen von Modul Komponenten ermöglichen
Beschreibung	In der Kursbearbeiten-Übersicht soll jede Komponente Question ein und ausklappbar sein. Dazu soll neben der Überschrift der jeweiligen Komponente eine Schaltfläche vorhanden sein. Ein Klick auf diese ermöglicht, abhängig vom aktuellen Zustand, das Ein- und Ausklappen.
Akzeptanzkriterium	Die Schaltfläche ist für jede Komponente vorhanden und durch das Klicken wird nur die betreffende Komponente animiert.
Story Points	8
Entwickler	ilhan
Umgesetzt in Iteration	18
Tatsächlicher Aufwand	7
Velocity	0.875 h/SP
Bemerkung	Ausgegliedert aus US 23

I Protokolle der Auftraggeber*innentreffen

Im folgenden sind alle Protokolle der Auftraggeber*innentreffen angehängt. Wir haben einen wöchentlichen Iterationszyklus angestrebt, allerdings einige Treffen wegen Feiertagen und anderen Umständen ausfallen lassen. Deswegen gibt es nicht für jede Iteration ein Protokoll. Die Abgabe der Userstories erfolgte in diesem Fall in einem kleineren Treffen, welches nicht protokolliert wurde, oder indem die Auftraggeber*innen die aktuelle Serverversion anschauten. Trotzdem wurden alle vorläufig abgenommenen Userstories auf dem nächsten offiziellen Treffen noch einmal komplett angeschaut.

Protokol Auftraggebertreffen 12.05.2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Claas A. Völcker
BP Clonecademy

Technische Fragen

- Multi device Nutzung
  - nur Desktop Nutzung
- Multilingualität
  - Fragen werden Kursen in einer Sprache zugeordnet, um mehrere Sprachen zu ermöglichen, muss der Kurs zweimal angelegt werden.
  - Das Interface wird mit Option auf Mehrsprachigkeit (Deutsch/Englisch) angelegt
- Die Webseite wird mit Bootstrap erstellt
- Die BP-Gruppe erhält keinen Admin Zugang zum Server des iGEM Teams

Inhaltliche Fragen

- Welche Kategorisierungen von Fragen gibt es?
  - Es gibt Kurse, die aus einer unterschiedlichen Menge an Fragen bestehen.
  - Die Kurse sind nach Schwierigkeit kategorisiert:Anfänger*innen Schule/Studium - Expert*innen
  - Jeder Kurs erhält optional Tags mit den Themen des Kurses (z.B.: Klonierungsverfahren)
- Kurse sind nach dem Anlegen nicht sofort sichtbar, sondern müssen erst freigeschaltet werden. Das erlaubt es, den Kurs weiter zu anzupassen, bevor Lernenden diesen bearbeiten.
- Alle Fragen bestehen aus mindestens einem Namen und gehören zu einem Kurs.
- Persistent gespeichert werden soll
  - Der Fortschritt eines*r Nutzer*in im Kurs
  - Die vollständig bearbeiteten Kurse
- Wie soll die Fragenseite aufgebaut sein?
  - Analog zu Codecademy: Links die Aufgabenstellung & weitere Informationen, Rechts das Bearbeitungsfenster
  - Beim Feedbackfenster wird Fragestellung & Antwortmöglichkeiten mit angezeigt
  - Ein Knopf zum Anzeigen eines Hinweises sollte eingebaut werden.
- Aufbau der Landing Page / des Dashboards
  - Wie bei Codecademy mit aktiven Kursen, Navigationsschaltflächen und weiteren Kursen
- Welche Nutzerrollen gibt es?
  - Identifiziert sind Admin (Zugriff auf alles), Moderator (stellt Kurse ein und gibt diese frei)
- Kein dynamisches Feedback, nur statische: Es war falsch, so ist der richtige Ansatz
- Kurse sollen nach Schwierigkeit und Kategorien filterbar sein
- Ob Nutzer*innen am Ende eines Kurses eine Möglichkeit zum Feedback gegeben wird, wird noch diskutiert.

Weiteres Vorgehen

- Ein PDF mit allen (vollständigen) Userstories (ToDo, Diese Iteration) immer bis Montag im Seafile hochladen

Protokoll 19.05.2017

CloneCadamy

TECHNISCHE UNIVERSITÄT DARMSTADT

Struktur der Kurse Beispiel:

- Grundlagen (Organisationsgruppe)
  - Einstieg I (? Frage)
  - Einstieg II (?)
  - PBS (Modul)
    - Translation (?)

- „Kurse“ sind öffentlich oder privat (private nicht einsehbar)
- Falls die Kurse privat sind, nur Einladung von Moderator
- Als Gruppe zusammen einen Kurs erstellen
  - mehrere Moderatoren können den Kurs bearbeiten
- Jede Organisation hat Moderator(en)
- Module mit Tags (variabel aber mit Vorgaben zur Auswahl)
  - Suchfunktion
- Module werden von Anfang bis zum Ende immer schwerer
- Fragen die für einen Kurs erstellt wurden, gehören dann auch nur zu diesem Kurs
  - Module die einem Kurs angehören können auch nur diesem Kurs angehören (keine Duplikate)
  - öffentliche Fragen sind immer öffentlich
- Nach jeder bearbeiteten Frage wird das Ergebnis in die Statistik eingetragen.
- Was haben Fragen gemeinsam?
  - Name, Inhalt, richtige Antwort, Feedback
  - Fragen Inhalt: Text mit oder ohne Bilder
- Feedback bekommt man am Ende vom Modul
- Man kann auch weiter wenn man eine Frage falsch beantwortet hat. Kann auch auf die falsch beantwortete Frage wieder zurück
- Bei falschen Antworten keine Hinweise und auch keine Lösung
- Dynamisches Feedback bei Fragen die sehr oft Falsch beantwortet werden(mit Default z.B. „FALSCH“ und Möglichkeit selber ein Feedback zu schreiben)

- Statistik für die Fragen
  - wie oft wurde die Frage Richtig oder Falsch beantwortet? Bsp:
    - Frage XY – XX (zweimal Falsch)
    - Frage YZ – XXXO (dreimal Falsch einmal richtig)
      - vom Lehrer einsehbar

Qualitätsmerkmale

- Sicherheit (Security)
  - Die Daten von Angreifer schützen
  - Nutzer Daten von Angreifer schützen
  - kein unbefugter Zugang in Admin/Mod Bereichen
- Veränderbarkeit (Update = einpflegen der Inhalte)
- Bedienbarkeit
  - MUSS einfach sein
- (Verbrauchsverhalten)

Protokol Auftraggebertreffen 2. Juni 2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Claas A. Völcker
BP Clonecademy

Technische Anmerkungen

- Server muss HTTPS sprechen
- Dockernutzer einrichten

Inhaltliche Anmerkungen

- Es geht weiter, sobald ein Nutzer die Frage richtig beantwortet hat.
  - Fehlerhighlighting an Fragentypus anpassen
- Lückentext können wir als kommende Frage aufnehmen
- Modulstruktur der Fragen
  - Lerngruppen fliegen raus
  - Nur zwei große Übergruppen: "iGEM", "allg. Biotech"
  - privat vs. öffentlich fliegt raus, nur sichtbar und nicht sichtbar bleibt (Entwurf und Öffentlich)
  - Datenbankverknüpfungen per ForeignKey
  - Fragen sind *immer* in Modulen
- Es wird eine Schaltfläche gewünscht: Moderator werden
  - Schaltfläche klicken, Mail an Admins, Freischalten oder nicht
- iGEM oder Biotech und Sprache wählbar machen (keine einmalige Wahl)
  - Sprache ist nur am Anfang auswählbar, damit man nicht mit den angebotenen Kursen durcheinander kommt
- Moderator kann einen Kurs nicht selbst freischalten, sondern muss die Freischaltung beim Admin beantragen
  - Wenn Kurse einmal freigeschaltet werden, können sie geändert werden
  - Wer kann Kurse später bearbeiten?
  - Kurse brauchen ein Feld: Wer hat angelegt
  - iGEM-Team berät sich zum Thema Moderations-Whitelist;
- Rechteentzug implementieren
- Lerntext
  - Infotext kommt wo, wird wie eingepflegt?
    - Wird im iGEM-Team diskutiert?

Qualitätssicherung

- Wie ist Änderbarkeit genau definiert?
  - Codemodifizierung einfach, Erweiterungen coden einfach
  - Wird auf Codeebene gewährleistet
    - Kommentare sind wichtig
    - Übersichtlicher, wartbarer Code
    - Hooks für Erweiterungen: gutes Vererbungsmodel

Weiteres Vorgehen

- Wir müssen genauer auf die konkrete Beschreibung in den User Stories achten
- Wir haben ein, zwei Kleinigkeiten vergessen
- Gestrichene Features verbleiben erst einmal im Code, damit sie später reingenommen werden können
- Das Wiki muss befüllt werden
  - Nutzerrechte werden im Development Wiki festgehalten
  - Dokumentation der API ist wichtig! Was wird übergeben auf welchen API Schnittstellen
- Code Refactoring vor allem im Backend

Protokol Auftraggebertreffen 9. Juni 2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Claas A. Völcker
BP Clonecademy

Anmerkung

- Das Protokoll ist ein Gedächtnisprotokoll, da durch einen Festplattenabsturz das Original verloren gegangen ist.
- Eventuelle Unstimmigkeiten bitte so schnell wie möglich melden.

User Stories

- Die präsentierten Userstories wurden abgenommen (siehe US PDF)
- ”Adminrechte freischalten” (ID 12) wurde in die nächste Iteration verlängert

Funktionalität

- Man kann Fragen überspringen und später beantworten
- Die Texte ändern sich nach Kurs und Modul, müssen also in beiden gespeichert werden

Aufbau der ersten Übersichtsseite

a) Auswahl der Sprache (Englisch/Deutsch), später nicht mehr änderbar

b) Auswahl der Kursgruppe: iGEM, Allgemein

Design

- Das grundlegende Design, dass von der BP Gruppe präsentiert wurde, ist als Basis gut
  - drei Farben, plus wieß und schwarz
  - Weiß, schwarz und grau sollten als Basisfarben verwendet werden, die anderen beiden als Akzente
  - (52/56/60), (196/6/51), (214/191/131)

Nutzerrollen

Die Namen der Rollen sind vorläufige Arbeitsversionen

- Nutzer: Können Fragen sehen und beantworten, können ihr eignes Profil einsehen
- Moderator*innen: Können Kurse neu anlegen und eigene Kurse ändern
- Trusted Moderator*innen: Können Kurse anlegen, die nicht mehr explizit freigeschaltet werden müssen, können alle Kurse ändern
- Admins: Können Rechte vergeben und entziehen, können Kurse freischalten
- Server-Admins: Haben Zugriff auf den Server und das Django Backend

Die Ordnung über die Gruppen impliziert, dass Nutzer*innen alle Rechte der kleineren Gruppe haben

- Zum Thema Accountlöschung/Blockierung tauscht sich das iGEM Team noch aus.

Weiteres Vorgehen

- Die kommende Iteration streckt sich über zwei Wochen
- Ziel der Iteration ist es, den existierenden Code zu überarbeiten, und Integrationstests durchzuführen, damit ein erster Release auf dem Server möglich ist
- Dazu wird das Design umgesetzt und die Datenbank sauber überarbeitet
- Alle existierenden Schnittstellen und der Code sollte dokumentiert sein
- Der Code wird auf Modularität überprüft
- Am 16.06. findet ein eigenes Treffen statt, um technische Fragen zu klären
- Als Ziel wird eine Vorstellung der Webseite auf dem iGEM Treffen angestrebt

Protokol Auftraggebertreffen 23. Juni 2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Claas A. Völcker
BP Clonecademy

Aufgabe 1 Bericht der letzte Iteration

Aufgabe 1.1 Design

Wir präsentieren das neue Design
Anmerkungen:

- Das Menü zur Nutzerauswahl ein Suchfeld haben und kein reines Dropdown sein

Aufgabe 1.2 Weitere Anmerkungen

- Kurskategorien sollten bei den Kursen auswählbar sein
  - z.B. via eines Reiters über der Übersicht im Dashboard
- Bei der Registrierung sollte die Lernsprache auswählbar sein
- Der "Get hint"Knopf fehlt noch
  - Wird in einer eigenen User Story programmiert
- Der Kursfortschritt ist noch nicht ganz klar
  - Das muss auf der Kursübersicht einsehbar sein (z.B. via Fortschrittsbalken)

Aufgabe 2 Vorschlag für neue User Stories:

- Nutzerdaten ändern
- Kurs fortführen
  - Knopf "Kurs starten"wird zu "Fortführen"
  - Man kann einen bereits begonnenen Kurs an jeder Stelle, an der man bereits war, wiederholt starten
- Moderatorenrechte vergeben
- Neue Fragen
- Sprachauswahl bei Registrierung
  - Klar machen, dass Englisch Primärsprache ist
  - Vorschlag: Anzahl der verfügbaren Kurse pro Sprache anzeigen

Aufgabe 2.1 Ausgewählt

- Moderatorenrechte vergeben
- Kurs fortführen
- Sprachauswahl

Aufgabe 3 Zielvereinbarung für die nächste Woche

- User Stories bearbeiten
- Refactoring erstmal abschließen
- Deployment ermöglichen

Aufgabe 4 Weitere Planung

- Vor einem iGEM Teamtreffen können Nutzerstudien gemacht werden
  - Am Anfang eine kleine Gruppe mit viel Beobachtung
  - Auf Leos Laptop, um Screencapture zu ermöglichen

Protokol Auftraggebertreffen 30. Juni 2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Claas A. Völcker
BP Clonecademy

Aufgabe 1 Bericht der letzte Iteration

Aufgabe 1.1 User Stories

- Kursfortschritt anzeigen
  - abgenommen
- Kurs fortsetzen
  - abgenommen
  - Anmerkungen: In den Nutzerstudien wurde bemängelt, dass nicht alle Buttons klar plaziert sind
  - Der start Course Button könnte die volle Kursübersicht ersetzen
- Sprache auswählen
  - abgenommen
  - die deutsche Lokalisierung fehlt noch zu Teilen

Aufgabe 2 Design Anmerkungen

- Die Fragenbeantwortung dritteln
  - 1. Drittel: Dashboard
  - 2. Drittel: Fragefenster
  - 3. Drittel: Feedbackfenster (wenn kein Feedback vorhanden, leer)
    - Visuelles Feedback zur Frage (z.B. durch Symbole)
    - Möglichkeit, einen Feedbacktext für korrekte Antworten einpflegen
    - Feedbacktext sollte Markdown Support haben
- Fragentitel sind erwünscht, wenn keine eingeben werden, sollte ein generischer aus dem Modulnamen generiert werden.

Aufgabe 3 Vorschlag für neue User Stories

- Drag & Drop
  - Anschauliche Plasmidkarten werden ins Seafile hochgeladen
  - Drag&Drop Elemente sollten nicht zu viel Informationen über die Lösung verraten
  - Beim Klick auf ein Element sollten die Lösungsmöglichkeiten klar ersichtlich sein
  - Es könnte mehrere richtige Lösungen geben, es wird überlegt, wie das eingepflegt werden kann
  - Vorschlag zum Einpflegen:
    - Mögliche Antwortfelder auswählbar
    - Mögliche "Stückchen" auswählbar
    - Gültige Kombinationsmöglichkeiten angebe
- Markdown Support

Aufgabe 4 Zielvereinbarung für die nächste Woche

- Es wird um eine realistische Einschätzung der Machbarkeit von Drag & Drop bis nächste Woche durch das BP Team gebeten
- BP Team diskutiert über die bessere Plazierung von Knöpfen
- Markdown Support wird angeschaut
- Überarbeitung des Designs der Frageseite ist erwünscht
- User Stories

Aufgabe 5 Weitere Planung

- Stabilitätstest wären cool

Protokol Auftraggebertreffen 14. Juli 2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Claas A. Völcker
BP Clonecademy

Aufgabe 1 Bericht der letzte Iteration

- Leo präsentiert das Frontend
  - Multiple-Choice zu Dritteln macht wenig Sinn, zu viel Platz wird für Feedback verschwendet
  - Der genaue Aufbau der Frageseite muss mit jedem Fragetypus neu evaluiert werden
  - Die Bestandteile der Frage könnten noch deutlicher getrennt werden
  - Vorschlag: Auf der "Kurs anlegen" Seite kann man fertig bearbeitete Komponenten wieder einklappen, damit man nicht zu viel Platz verschwendet → wird eine User Story

Aufgabe 1.1 User Stories

- "Moderationsrechte freischalten"
  - Feedback zur Seite ist noch nicht vorhanden
  - nicht abgenommen

Aufgabe 2 Konkretisierte User Stories

- Wir beschränken uns auf Plasmidkarten
  - Es gibt eine kreisförmige Schablone, mit n Feldern
  - Diese wird vorher implementiert und dann beim Erstellen ausgewählt
- Zwischen 3 und 4 Elementen auf einen Kreis
- Welche Antwortbausteine gibt es?
  - Auch falsche, die gibt der Aufgabensteller ein
  - Es gibt einen Satz an vorgefertigten Elementen, die man beim Eintragen beschriften kann

Aufgabe 3 Zielvereinbarung für die nächste Woche

- Das Refactoring wird beendet

Aufgabe 3.1 Neue User Stories

- Drag And Drop (Plasmidkarten) (Moderator/Nutzer)
- Nutzerdetails ändern (Nutzer)
- Neue Kategorie hinzufügen (Administrator)
- Übersicht über das eigene Profil (Nutzer)

Aufgabe 4 Weitere Planung

- Das iGEM Team plant einen Release Termin, der passt.
- Dafür werden die Server-Leute gebraucht.
- Es wird ein Termin um das erste August Wochenende angestrebt.

Protokol Auftraggebertreffen 21. Juli 2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Leonhard Wiedmann
BP Clonecademy

Aufgabe 1 Bericht der letzte Iteration

- Die Userstorys werden vorgezeigt
  - bei edit course sollen
  - us-15(Passwort und email ändern) wurde abgenommen

Aufgabe 1.1 User Stories

- ”edit course”
  - bessere Fehlermeldungen wenn Felder leer
- ”change password”
  - done
- ”change email”
  - done

Aufgabe 2 Konkretisierte User Stories

- Plasmidkarten wurden wieder verworfen
- Es wird zuerst die Multiple choice Frage überarbeitet

Aufgabe 2.1 ”Multiple choice Fragen”

- Markdown prüfen ob man es erweitern kann (speziell Pfeile um Text)
- Bild zur Frage hinzufügen
- Bild zum Feedback hinzufügen
- Bild zu möglichen Antworten hinzufügen

Aufgabe 3 Zielvereinbarung für die nächste Woche

- Das Design wird überarbeitet
- Die Mutiple Choice Frage wird überarbeitet

Aufgabe 3.1 Design

- der ganze Bildschirm soll ausgefüllt sein
- den grauen Hintergrund entfernen
- hover Effekt der Buttons überarbeiten
- header kleiner machen
- Schattierungen in der Hauptbox entfernen

Aufgabe 4 Weitere Planung

Aufgabe 4.1 ”visibility”

- Ein Moderator kann alle invisible Kurse sehen
- ein Moderator kann auch Kurse invisible schalten
- der Admin kann auch Kurse löschen
- der Trusted Moderator hat die gleichen rechte wie ein Admin im Kurs Bereich

Protokoll Auftraggebertreffen 28. Juli 2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Claas Völcker
BP Clonecademy

Bericht der letzte Iteration

User Stories

- bei den Nutzerprofil Änderungen wird die Passwortbestätigung nicht richtig überprüft
- Kurs bearbeiten: die Fehlermeldung wird nicht korrekt angezeigt
- Markdown: angerissen, nicht fertig
- Bilder hochladen: abgenommen
  - kleine Anmerkung: die Zwischenüberschriften "Image for the question" und "Image for the result" ist redundant

Design

- gewählte Schriftart: Railway (von Google)
- finaler Farbcode aus Sample im Seafile
  - Achtung: weiß ist eigentlich ein Grauton, bitte beachten
- Logo im Seafile: full_red_blue_Main
- die Akzentfarbe Blau wird den bisherigen Sandton ersetzten, die Buttons werden Grau, das BP Team setzt ein paar Kombinationsmöglichkeiten um
- in einem eigenen Treffen kann das Design dann weiter verfeinert werden

Kommende Iteration

- kommende Woche soll das Deployment stattfinden
  - 7. August, 14:00 Uhr, im Schreibraum des iGEM Team, Claas schließt sich mit Markus kurz
- Design-Treffen: Dienstag, 14:00 Uhr, an der Biologie Seminarraum Botanik, bis dahin werden die Farben eingearbeitet
- Skalierung der hochgeladenen Bilder muss bedacht werden

Weitere Planung

- YouTube Embedding wurde angesprochen
  - wurde vom BP Team als machbar eingestuft

Protokoll Auftraggebertreffen 21. August 2017

Claas Völcker
BP Clonecademy

Bericht der letzte Iteration

User Stories

- bei den Nutzerprofil Änderungen wird die Passwortbestätigung nun korrekt überprüft
- Kurs bearbeiten: Bild hochladen funktioniert zur Zeit nur als .jpg außerdem wird das Bild nicht angezeigt

Deployment

- Das beste wäre, wenn die WebSite am Montag aus dem Internet erreichbar wäre.
- Wir merken an, dass wir keine Sicherheit garntieren können.
- Als größte Gefahr wird ein (D)DoS eingeschätzt, der möglicherweise nur die VM abschießt, und nicht die gesamte Serverhardware lahmlegt.

Wettbewerbsjury

- Beim iGem Wettbewerb werden Juror*innen wenig Zeit haben sich die WebSite anzusehen.
- Es wird angestrebt eine leicht zugängliche Demo auch ohne Log-in bereit zu stellen.

Kommende Iteration

- Bilder: Müssen bis Montag angezeigt werden
- Infofragetyp: Nur Fragetext, keine Antwortmögli chkeiten, nur auf weiter drücken
- Kurs bearbeiten Seite: soll "übersichtlicher"werden: Module und andere Elemente sollen einklappbar sein
- Introseite: statische HTML Seite, die Hardgecoded auf dem Server liegt.
- Fragen beantworten: Navigation zwischen Fragen in einem Kurs soll möglich sein.

Weitere Planung

-
- Es wird angestrebt bis Ende August alle relevanten Features einzubringen und nach der zweiten Septemberwoche definitiv keine weiteren Features zu implementieren.

Protokoll Auftraggebertreffen 21. August 2017

Claas Völcker
BP Clonecademy

User Stories

- Info Fragetyp einpflegen - abgenommen
- Info Fragetyp anzeigen - abgenommen
  - Bild sollte 2/3 der Seite einnehmen, der Text kommt auf die rechte Seite (siehe Designänderungen).
- Static Pages - abgenommen
  - Der Knopf in der Kopfleiste wird direkt auf die "About" Seite verlinken, die anderen statischen Seiten werden in der Fußleiste verlinkt.
  - Es soll folgende Seiten geben "Privacy Policy", "Terms", "Contact", "About", "Impressum"
- Statistik exportieren - abgenommen
- Dashboard im Kurs anzeigen - abgenommen
  - Kann größer gemacht werden (1/3 der Seite)
  - Weitere US:
    - Knopf der aufs Kursübergreifende Dashboard führt ("Zurück" Icon)
    - Daneben: Moderator Name mit Profil
    - Daneben: Dashboard schließen
    - Kurs, Kursbeschreibung, Modultitel, Modulbeschreibung als ausklappbares Feld, Fragentitel (mit Navigations-Knöpfen)
    - Aktuelle Frage sollte auch hervorgehoben werden
    - Dashboard Knopf kleiner
- Zusätzliche Funktionalität: YouTube Videos
  - YouTube Videos können anstelle des Bildes in einen InfoText eingebaut werden.
  - Am Ende soll es die Möglichkeit geben, zwei Fragetypen auszuwählen "InfoText (YouTube)" und "InfoText (Bild)"
- Antwortmöglichkeiten Bilder einpflegen - abgenommen
  - In der Fragenbeantwortung sollten die Antwortbilder größer sein

Änderungen & Neue Features

Alle Änderungen finden sich en detail handschriftlich im Seafile!

- Navbar (von links nach rechts)
  - About/information
  - Learn
  - Profile
- Fragen Anzeigen
  - Fragebeschreibung und Fragestellung in zwei verschiedenen Boxen anzeigen
  - Daraus resultiert, dass es Fragentitel, Fragentext und Fragestellung gibt
  - Kursbeschreibung soll hinzugefügt werden
  - Kursbeschreibung muss eine Zeichenbeschränkung erhalten (144 Zeichen?)
- Fußleiste in einer Frage
  a) Dashboard (umbenannt zu Overview)
  b) Fragetitel
  c) Vor-Zurück Pfeile (in der Mitte Fragenummer/Anzahl von Fragen)
- Fußleiste außerhalb
  - Static Page Buttons
  - Copyright & Siteinformation (Name)
  - Social Media Accounts der iGEM Gruppe
  - Farbe: Wie Header

- Profile
  - Dropdown aus dem Knopf
  - Nutzerbild (default wäre z.B. stylisierter DNA Strang)
  - Punktzahl (siehe Achievements)
  - ”View Profile” Schaltfläche
  - Logout
- Statistiken
  - Vom aktuellen Standpunkt aus 7 Tage zurück: Wie viele Fragen wurden beantwortet
  - Kreisdiagramm Aufteilung auf Kategorien
    - Kurse müssen Farben haben
- Quiz
  - Nur die mittlere Spalte (also Antwortmöglichkeiten)
  - Nur die Fragestellungen werden angezeigt, keine Texte und Beschreibungen
  - Auswertung wie gehabt
  - Keine Möglichkeit zur Navigation
  - Jeder Kurs hat optional ein Quiz (min. 10 Fragen) am Ende
  - Optional ein Bild anzeigen
  - Die Antwort einer einmal beantwortete Quizfrage kann nicht nochmal ge’ndert werden, es sei denn man startet das Quiz von vorne
- Vorschläge der BP Gruppe
  - Password recovery in Profile or Login
  - Moderatorenrechte entziehen
    - Die Kurse können dann vorläufig nur noch von Admins bearbeitet werden
  - Adminrechte vergeben oder entziehen
  - Kategorien bearbeiten für Admins
    - Kategorien löschen benötigt eine Warnung, da alle Kurse in dieser Kategorie mit gelöscht werden.
  - Übersicht über die Kurse auf der Frontpage
    - Kurstitel und -beschreibung
    - aktuelles Modul
    - Knopf zum Fortführen
    - Fortschrittsanzeige
  - In Sidenav Moderatorenkurse anzeigen

Kommende Iteration

- Footer Redesign
- Besprochene Änderungen und Redesign (Dashboard, Infotext)
- Password recovery im Login
- Moderatorenrechte entziehen
- Adminrechte vergeben oder entziehen
- Kategorien bearbeiten für Admins
- ”Learn” Seite Übersicht
- Statistik

Protokoll Auftraggebertreffen 1. September 2017

Claas Völcker
BP Clonecademy

Bericht der letzte Iteration

User Stories

- US 30: abgenommen
  - Sterne sollten auch an anderen Stellen verwendet werden
- US 31: abgenommen
- US 36: abgenommen
- US 37: abgenommen
  - Statische Seite "Privacy Policy" sollte eventuell noch hinzugefügt werden
  - Trennstriche zwischen den statischen Links
  - externe Links einfarbig (nur weise Silhouetten)
  - Instagramm fehlt
  - "Dashboard" Button wird in "Overview" umbenannt
- US 40: abgenommen
- US 41: abgenommen

User Stories - in progress

- Modulbeschreibung sollte im Sidenav nicht auftauchen und auch nicht zeichenbegrenzt sein
- Module sollten im Sidenav ein- und ausklappbar sein
- Fragenanzahl in der "Kurs bearbeiten" Übersicht ist eine gute Idee

Anmerkungen und Verbesserungen

- Custom Feedback wird im Popup Fenster angezeigt

Kommende Iteration

- Nutzerstudie 06.09. um 13:00 Uhr
- Serverdeployment
- weitere Altlasten aufarbeiten
- Quiz

Protokoll Auftraggebertreffen 08.09.2017

Leonhard Wiedmann
BP Clonecademy

Bericht der letzte Iteration

User Stories

- US 42 abgenommen
  - alle buttons oben außer x entfernen und x rechts alignen
- US 43: abgenommen
- US 44: abgenommen
- US 46: abgenommen
  - User kann sich unsichtbar im Ranking setzten
- US 31: abgenommen
- US 34: abgenommen
- US 33: abgenommen
  - farbe ändern wird eigene userstory (wenn colorpicker funktioniert)

Fragen abschluss

- nach dem Quiz kommt ein Fenster mit feedback. Hier steht eine liste mit den richtgen und Falsch beantworteten Fragen.
- Feedback nach Fragen beantworten gibt ein Popup mit Feedback

Curse löschen/invisible

- in der Kursübersicht kommt ein Button für visible/invisible (offenes/geschlossenes Auge). Der Button von geschlossen zu offen kann nur von Moderatoren benutzt werden, wobei der von offen zu geschlossen auch vom responsible Mod verwendet werden kann.
- in der Kursübersicht kommt ein Button für Admins, wodurch man den Kurs löschen kann

Protokoll Auftraggebertreffen 15.09.2017

Claas Völcker
BP Clonecademy

Bericht der letzte Iteration

User Stories

- US 51 Profilbild hochladen: abgenommen
  - Anmerkung: Ein PopUp/Hinweis, der anzeigt, dass man sich noch einmal authentifizieren muss, wäre hilfreich
  - neue Userstory: Änderungen im Profil autorisieren (z.B. in diesem PopUp altes Passwort eingeben)
- US 49 Profile Quickview: abgenommen
- US 45 Eigenes Ranking anzeigen: abgenommen
  - Der Schlüssel für die Punkte im Ranking muss angepasst werden.
  - Nach n richtigen Fragen gibt es Zusatzpunkte.
  - Diese Zusatzpunkte dürfen nicht doppelt vergeben werden.
- US 39 Dashboard Moderator Übersicht: abgenommen
- US 50 Anonyme Statistik: abgenommen
- US 54 Visible schalten: abgenommen
- US 55 Invisible schalten: abgenommen
- US 47 Detaillierte Persönliche Statistik: abgenommen

Weitere Planung - Neue User Stories

a) Profiländerungen authentifizieren: siehe oben - Tobias

b) Persönliche Statistik erweitern um gesamt Fragen richtig/gesamt und eine Legende - Leo

c) Quiz Feedback: Dies sollte direkt nach jeder Frage angezeigt werden. - Claas

d) Quiz Fragen und Antworten: Reihenfolge randomisieren - Leo

e) Popup bei Fragenantwort anzeigen - Tobias

f) Ranking Berechnung: siehe oben - Claas

Weitere Ideen

- Profile Page - Ilhan
- Started Courses links in der Liste anzeigen - Ilhan
- Kategoriestatistik nur richtig beantwortete Fragen anzeigen - wurde gerade repariert
- Colorpicker - todo/offen

Noch im Backlog

- Kurs ändern übersichtlicher machen
- Passwort Reset anfordern
- Kursstatistik für Moderatoren
- Update auf den Server schieben (404 Bug finden)

Protokoll Auftraggebertreffen 15.09.2017

Claas Völcker
BP Clonecademy

Bericht der letzte Iteration

User Stories

- US 48: Kursstatistik anzeigen, abgenommen
  - Quiz Statistik ist noch nicht vorhanden
- US 52: Quiz Feedback anzeigen, abgenommen
- US 29: Passwort anfordern, abgenommen
- US 23: Kurs Edit übersichtlicher machen, abgenommen
  - Die Animationen hacken noch, es wurde eine Issue aufgemacht

Weitere Planung - Neue User Stories

- Letzte Iteration werden keine User Stories mehr gemacht
- Montag, 25.09. 14:30 Uhr Nutzerstudie (Abends ist Grillen)
- Freitag, 29.09., letztes Auftraggeber*innengespräch
- Freitag, 06.10. 20:00 Uhr, Vorstellung Clonecademy in großer Runde

Noch im Backlog

- Update auf den Server schieben (404 Bug finden)

- Die Entwickler einigen sich mit den Auftraggeber*innen darauf, dass die Software unter der MIT Lizenz veröffentlicht wird. Urheber im Sinn dieser Lizenz sind die vier Mitglieder des Entwicklungsteams.

  Copyright (c) 2017 Tobias Huber, Ilhan Simsiki, Claas Völcker, Leonhard Wiedmann

  Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

  The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

- keine weiteren Einträge

K Endgültiger Stand des Projektes und Übergabe

K.1 Stand des Projektes bei Abgabe

Zum letzten Auftraggeber*innentreffen hatte die Plattform *Clonecademy* alle vereinbarten Funktionalitäten. Die zu Beginn des Dokumentes skizzierten Ziele wurden somit erreicht.

Über die angestrebten Grundziele hinaus hat sich das Entwicklungsteam zusammen mit den Auftraggeber*innen dafür entschieden, vor allem eine umfassende Nutzer*innenverwaltung umzusetzen. Dadurch können nun die Gruppenzugehörigkeiten aller Accounts verändert werden und Nutzer*innen haben diverse Möglichkeit ihre Registrierungsdetails zu ändern.

K.2 Vereinbarungen zur Übergabe des Projektes

Der Sourcecode des Projektes wird auf Github unter der MIT Lizenz veröffentlicht. Während der Entwicklung des Projektes war das zugehörige Repositoy nur für die Entwickler und die Auftraggeber*innen einsehbar. Nach offiziellem Ende des Bachelorprojektes wird das Repository in ein öffentliches umgewandelt und steht damit allen Entwicklern im Netz zur Verfügung. Dies entspricht sowohl den Wünschen des Teams und der Auftraggeber*innen, als auch den Anforderungen des iGEM Wettbewerbs.

Das ursprüngliche Repository wird weiterhin von Claas Völcker verwaltet. Eine fortlaufende Entwicklung durch das Team, aber auch durch interessierte Dritte ist damit möglich. Leonhard Wiedmann hat bereits sein Interesse angekündigt, die Webseite weiter mit dem iGEM-Team zusammen zu betreuen.

Allen Entwicklern des Teams wird im Rahmen der Teampräsentation des iGEM Teams gedankt.

L Abschließende Bemerkungen des Teams zum Projekt