OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

Ostbayerische Technische Hochschule Regensburg

Fakultät für Informatik

Bachelorarbeit

im Studiengang Informatik

zur Erlangung des akademischen Grades
Bachelor of Science

Thema: Analyse, Konzeption und Implementierung eines Tools für das automatisierte Testen, Einreichen und Benoten von Programmieraufgaben

Autor: Andreas Huber <andreas.huber@st.oth-regensburg.de>
Matr.-Nr. 3180161

Version vom: 14. Februar 2022

Betreuer: Prof. Dr. Markus Heckner

Inhaltsverzeichnis

1 Einleitung und Motivation **1**

1.1 Herausforderung digitales Lernen 1

1.2 Kurs Digital Skills als Zusatzstudium 1

1.3 Struktur der Arbeit 1

2 Anforderungsanalyse **1**

2.1 Funktionale Anforderungen 1

2.1.1 Studierende 1

2.1.2 Lehrende 2

2.2 Nichtfunktionale Anforderungen 2

3 Softwarearchitektur **4**

3.1 Vergleich vorhandener Systeme 4

3.1.1 CS50 der Harvard University 4

3.1.2 Code FREAK der Fachhochschule Kiel 7

3.1.3 GitHub Classroom 9

3.2 Finale Architektur Online-Learning Platform 11

3.2.1 Tutors als Aufgabensammlung 11

3.2.2 GitHub Classroom als Abgabe- und Bewertungssystem 11

3.2.3 Replit als Online-Entwicklungsumgebung 11

4 Konfiguration und Implementierung **12**

4.1 Tutors als Aufgabensammlung 12

4.2 GitHub Classroom 12

4.2.1 Konfiguration 12
4.2.2 Erste Aufgaben 12
4.2.3 Tests und Benotung 12
4.3 Erstellung eines Replit-Starter-Templates 13
4.3.1 Template-Repository 13
4.3.2 Wrapper-Tools (get, check, submit) 13
4.3.3 OTH-Console 14
4.3.4 SSH-Keys 15

5 Studie: Test an fachfremden Studierenden 15

6 Zusammenfassung und Ausblick 15

Literaturverzeichnis iv

1 Einleitung und Motivation

1.1 Herausforderung digitales Lernen

1.2 Kurs Digital Skills als Zusatzstudium

1.3 Struktur der Arbeit

Die Arbeit besteht aus mehreren wesentlichen Bestandteilen. Kapitel 2 behandelt alle nötigen Anforderungen, die für den Aufbau einer digitalen Lernplattform wichtig sind. Kapitel 3 wiederum vergleicht und diskutiert zuerst vorhandene Lern-Systeme und trifft dann die finale Entscheidung für den Kurs Digital Skills. Die finale Entscheidung und deren Konfiguration und Implementierung wird dann folgend in Kapitel 4 näher erläutert und beschrieben. Schließlich wird die Programmierlernplattform in Kapitel 5 in Form einer kleinen Studie an nicht Informatik- oder Mathematikstudierenden Testprobanden getestet. Schließlich fasst Kapitel 6 alles zusammen und gibt einen weiteren Ausblick auf die Zukunft des Systems.

2 Anforderungsanalyse

Funktionale Anforderungen beschreiben, was das Projekt tun soll. Diese Informationen sind wichtig, um die richtigen Werkzeuge und Programme für die Umsetzung auszuwählen. Nichtfunktionale Bedingungen sind Bedingungen, wie zum Beispiel Zuverlässigkeit, oder diverse Sicherheitsanforderungen.

2.1 Funktionale Anforderungen

2.1.1 Studierende

Studierende sollen eine Kursübersicht mit allen Aufgaben haben. Der Fortschritt von jeweiligen Aufgaben sollte dabei leicht ersichtlich sein. Mögliche Deadlines oder Abgabefristen sollen bei jeder Aufgabe deutlich erkennbar sein.

Des Weiteren müssen Studierende die Möglichkeit haben, ohne die Installation von zusätzlichen Programmen die Aufgaben online bearbeiten, prüfen und abgeben

zu können. Trotzdem sollen sie die Option haben, die Aufgaben in der Entwicklungsumgebung ihrer Wahl lösen zu können.

Eine weitere wichtige Anforderung bei der Prüfung der Aufgaben ist, dass die Bearbeiter der Aufgaben sogenannte „human-readable“-Fehlermeldungen erhalten müssen. Das bedeutet, dass die Fehlermeldungen bei der Überprüfung auch für nicht-technische Studenten leicht verständlich sein müssen. Fehlermeldungen bzw. konstruktives Feedback muss dabei automatisiert und jederzeit generiert werden können.

Die Möglichkeit Aufgabenversuche abzugeben, muss ebenfalls mit wenig Aufwand behaftet sein. Falls ein Versuch fehlschlägt, oder nicht die volle Punktzahl erhält, sollte der Student jederzeit einen neuen Versuch hochladen können.

2.1.2 Lehrende

Lehrende müssen neue Aufgaben anlegen und konfigurieren können, dazu gehört unter anderem eine Deadline bzw. ein Abgabedatum. Des Weiteren sollen Lehrer eine Übersicht an Aufgaben des jeweiligen Kurses haben. Außerdem sollten Lehrende einzelne Aufgaben temporär verstecken oder deaktivieren können.

Überdies hinaus muss es möglich sein, mehrere Administratoren zu den Kursen hinzuzufügen. Dadurch ist es möglich, dass verschiedene Personen die Aufgaben erstellen und die abgegebenen Lösungen herunterladen können. Dies ist gleichzeitig die nächste Anforderung: Administratoren müssen mit wenig Aufwand alle Abgaben der Teilnehmer herunterladen können.

Ferner müssen die Aufgaben und die jeweiligen Deadlines auch nach der Erstellung editierbar sein.

Lehrende müssen den Aufgaben-Fortschritt der Teilnehmer je nach Kurs übersichtlich einsehen können.

2.2 Nichtfunktionale Anforderungen

Das eingesetzte System muss neben den funktionalen Anforderungen auch diverse nichtfunktionale Anforderungen erfüllen, um von der OTH als sinnvolle Lernplattform eingesetzt werden zu können.

Der erste wichtige Punkt ist, dass die Lernplattform möglichst zuverlässig ist.

Zur Zuverlässigkeit gehört neben einer hohen Verfügbarkeit auch eine skalierende Performance, wenn viele Studierende gleichzeitig die Plattform nutzen wollen.

Ein weiterer Punkt ist die Wartbarkeit. Die Plattform sollte mit möglichst wenig Wartung sicher bestehen bleiben können. Außerdem sollte nur auf externe Systeme gesetzt werden, von denen ausgegangen werden kann, dass diese noch einige Jahre gepflegt werden. Hier empfiehlt sich ein Aufteilen der Plattform auf mehrere externe Tools, um bei einem Ausfall oder Außerbetriebnahme eines einzelnen Tools noch einen Notbetrieb gewährleisten zu können. Der Austausch gegen eine andere neue Softwarekomponente gestaltet sich dadurch leichter.

Zu guter Letzt spielt die Ressourcenlast eine weitere Rolle bei der Entscheidungsfindung. Wenn Teile der Lernplattform auf OTH-Servern gehostet werden müssen, sollten diese möglichst ressourcenschonend sein. Serverressourcen sind teuer und können das Projekt im Zweifelsfall unrentabel machen, wenn das System bei paralleler Nutzung durch mehrere Studierende eine inadäquate Serverlast voraussetzen würde.

3 Softwarearchitektur

Dieses Kapitel befasst sich mit der für das Projekt benötigten Toolchain. Eine Toolchain ist eine Sammlung verschiedener Anwendungen, die gemeinsam eine Lösung bzw. ein Produkt erzeugen. Durch den Vergleich mit verschiedenen bestehenden digitalen Lernplattformen ist es möglich, optimale Software-Werkzeuge für die Hochschule Regensburg zu finden und schließlich einzusetzen.

3.1 Vergleich vorhandener Systeme

3.1.1 CS50 der Harvard University

3.1.1.1 Allgemeines

CS50 ist die ursprüngliche Bezeichnung eines Lernkurses über Informatik, welcher von der Harvard University ins Leben gerufen wurde und weiterhin betreut wird. Der Kurs wurde aufgrund seines Erfolgs digitalisiert und wird nun als CS50x auf der Lernplattform edX angeboten. Folgende Recherchen und Aussagen sind jeweils immer auf die Online-Version CS50x bezogen.

Der Kurs CS50 lehrt Schüler[1] die Grundlagen der Informatik. Dabei werden diverse Programmierübungen abgefragt. Aufgrund der hohen Anzahl an Teilnehmern besitzt der Kurs ein automatisiertes Abgabe- und Benotungssystem.

Das System hinter CS50 wird mittlerweile vielfältig eingesetzt und wurde zu einem universalen Online-Lernsystem erweitert. Jeder kann sich durch eine Authentifizierung über die Plattform GitHub im Abgabesystem von CS50 einloggen und eigene Kurse erstellen. (Harvard University, n. d. b)

[1] Aus Gründen der besseren Lesbarkeit wird in der gesamten Arbeit auf die gleichzeitige Verwendung der Sprachformen männlich, weiblich und divers (m/w/d) verzichtet. Sämtliche Personenbezeichnungen gelten gleichermaßen für alle Geschlechter. Ist eine spezifische Geschlechtergruppe gemeint, wird das entsprechende Adjektiv vorangestellt (z.B. „männliche Studenten“)

3.1.1.2 Ablauf für Studierende

Dem Teilnehmer wird jede Woche ein neues Kapitel präsentiert. Er kann sich dabei sowohl durch ein Vorlesungsvideo, als auch durch geschriebene Materialien über das Thema der Woche informieren. Mit Beginn der Woche bekommt der Teilnehmer neben den Materialien auch Programmieraufgaben, welche er mit dem vorher genannten System bearbeiten kann. (Harvard University, n. d. c)

Die Programmieraufgaben können wahlweise über die, auf AWS Cloud9 basierenden, Online-Entwicklungsumgebung „CS50-IDE“ von Harvard oder in jeder anderen beliebigen Entwicklungsumgebung der Wahl bearbeitet werden. Dies wird durch die Architektur des Systems ermöglicht. Jede Funktionalität der Automatisierung geschieht durch Kommandozeilen-Tools. Dieses System hat den Vorteil, dass es unabhängig von der eingesetzten IDE funktioniert, es wird lediglich ein Terminal mit den jeweiligen Tools benötigt. (Harvard University, n. d. d)

Vor der Abgabe der endgültigen Lösung mit dem sogenannten Werkzeug „submit50“, ist es möglich den Code mit einem weiteren Werkzeug namens „check50“ überprüfen zu lassen. Außerdem gibt es viele weitere Werkzeuge, ein Beispiel hierfür ist „style50“, welches die Qualität und den Style des Programmcodes überprüft und bewertet. (Harvard University, n. d. e)

3.1.1.3 Architektur

Die Harvard University hält den Aufbau von CS50 weitestgehend transparent. Viele der eingesetzten Werkzeuge sind öffentlich als Open-Source-Projekte unter der GitHub-Organisation „CS50“ zu finden (Harvard University, n. d. a). Darunter befinden sich unter anderem folgende Projekte:

- submit50: Abgabe von Code
- check50: Funktionalitätstests des Codes
- render50: Erzeugung von .PDF-Dateien aus Code
- ide50: Online-Entwicklungsumgebung
- style50: Überprüfung der Code-Qualität
- compare50: Plagiatserkennung von abgegebenen Projekten
- server50: Webserver

In Abbildung 1 ist der Aufbau der Lernplattform CS50 vereinfacht grafisch dargestellt. Die Rechtecke und Menschen sind die beteiligten Komponenten. Die Pfeile zwischen den Komponenten beschreiben die verbindende Relation. Von der „CS50-Programm“-Komponente ausgehende gestrichelte Pfeile zeigen, je nach ausgeführtem Befehl, mögliche Relationen.

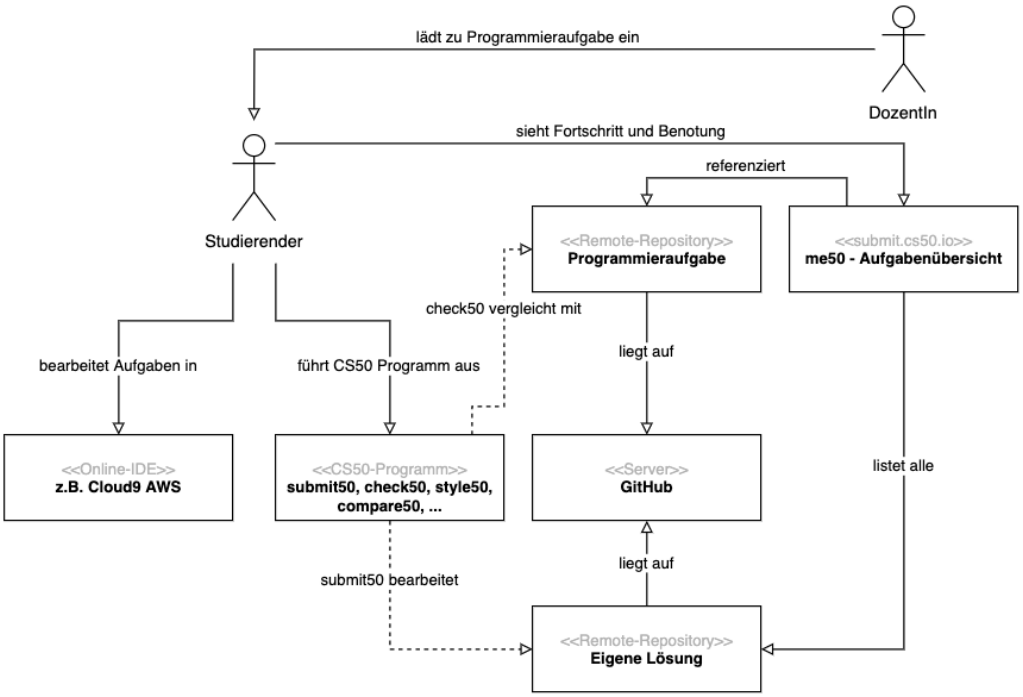

Abbildung 1: CS50 Architektur
Quelle: Eigene Darstellung

3.1.1.4 Probleme beim Einsatz für die OTH

Die Toolchain von CS50 wäre adäquat für den Einsatz an der OTH-Regensburg. Eines der Projekte ist aktuell noch nicht Open-Source: die Website zur Erstellung von neuen Kursen, Abgaben und Mitgliederverwaltung. Dieses Projekt ist essenziell für die Verwendung der Werkzeuge an der Regensburger Hochschule. Nach Rücksprache mit Herrn Carter Zenke der Harvard University ist ein Neuaufbau dieser Website mit einhergehender Veröffentlichung als Open-Source-Projekt gerade in Planung. Einen genauen öffentlichen Zeitplan hierfür gibt es aktuell nicht. Infolgedessen ist ein Einsatz des CS50-Systems an der OTH zum heutigen Datum nicht möglich.

3.1.2 Code FREAK der Fachhochschule Kiel

3.1.2.1 Allgemeines

Code FREAK ist eine All-in-one-Lösung für Online-Programmieraufgaben mit automatisiertem Feedback. Die von der Hochschule Kiel entwickelte Open-Source-Software soll den Einstieg in eine digitale Lernumgebung einfach machen. Die Zielgruppe von Code FREAK bezieht sich hierbei explizit auf Universitäten und Einrichtungen einer höheren Bildung. (Kiel University of Applied Sciences, 2019-2020)

Des Weiteren wirbt Code FREAK mit einer LMS-Integration. LMS ist die englische Abkürzung für Learning Management System (dt. Lernplattform). Viele Schulen und Universitäten verwenden bekannte LMS-Systeme, wie Moodle, um Kurse, Fächer und Studierende zu verwalten (moodle.de, n. d.). Dies hat den Vorteil, dass Studierende kein extra Nutzerkonto für Code FREAK anlegen müssen. Sie können sich direkt mit ihrem gewohnten Zugang anmelden.

3.1.2.2 Ablauf für Studierende

Studierende können nach Anmeldung am System, an für sie sichtbaren Kursen (Assignments) teilnehmen. Dies geschieht meist durch einen vom Dozenten generierten Einladungslink. Jedes Assignment kann mehrere Aufgaben (Tasks) enthalten. Die einzelnen Aufgaben können dann entweder über die integrierte Online-Entwicklungsumgebung, durch Hochladen der Lösung, oder durch die Angabe eines Git-Repository-Links bearbeitet werden.

Mit dem Klick auf den Knopf „Start Evaluation“ wird hochgeladene Lösung der Aufgabe überprüft. Die Ergebnisse der einzelnen Test-Schritte können dann in einem weiteren Tab eingesehen werden. Wenn alle Tests der Aufgabe bestanden wurden, wird der Task mit einem grünen Haken versehen. So sieht der Schüler in der Assignment-Ansicht auf einen Blick, welche Aufgaben er bereits gelöst hat.

3.1.2.3 Architektur

Code FREAK wird als fertiger Docker-Container ausgeliefert. Docker ist eine Software um mithilfe von Containervirtualisierung einzelne Anwendungen voneinander zu isolieren. Durch die Containervirtualisierung werden unter anderem viele Abhängigkeits- und Einrichtungsprobleme beseitigt. Der Container enthält alle Bibliotheken und Abhängigkeiten, die die Software für die Laufzeit benötigt (Mouat,

2015). Docker entstand auf Basis von Linux-Containern, welche ein oder mehrere Prozesse vom restlichen System isolieren können (Red Hat, 2018).

Durch diese Handhabung kann Code FREAK mit nur einem Befehl in der Kommandozeile installiert und gestartet werden. Die Software benötigt grundsätzlich nur einen Container. Benutzt ein Studierender jedoch die integrierte Online-Entwicklungsumgebung (Online-IDE), muss für jede Instanz ein zusätzlicher Container mit der Laufzeitumgebung der IDE gestartet werden. Jeder weitere Container benötigt weiteren Arbeitsspeicher.

3.1.2.4 Probleme beim Einsatz für die OTH

Beim Einsatz von Code FREAK an der Hochschule Regensburg gibt es einige Probleme. Das erste Problem bezieht sich auf den vorher erwähnten Arbeitsspeicher. Die Praxis zeigt, dass eine Instanz der Online-IDE schon nach wenigen Dateien 3 Gigabytes an Arbeitsspeicher verwendet. Um eine reibungslose und parallele Nutzung für alle Studierende des Kurses gewährleisten zu können, werden dementsprechend sehr hohe Serverkosten fällig. (Kiel University of Applied Sciences, n. d. b)

Eine weitere Hürde ist die Stabilität der Software. Code FREAK befindet sich, Stand heute, mitten in der Entwicklung, weshalb einige Funktionen und Features noch nicht ordnungsgemäß funktionieren. Darunter die vorher geworbene LMS-Integration (Kiel University of Applied Sciences, n. d. a). Zum heutigen Zeitpunkt ist LDAP die einzige Möglichkeit sich mit dem System zu authentifizieren. Ein eigenes Anmeldeverfahren gibt es nicht.

LDAP steht für „Lightweight Directory Access Protocol“ und ist ein Netzwerkprotokoll zur Durchführung von Abfragen und Änderungen in einem verteilten Verzeichnisdienst. LDAP ist der De-facto Industriestandard für Authentifizierung und Autorisierung. (LDAP, n. d.)

Der Kurs Digital Skills soll den Teilnehmern einen Überblick über den Arbeitsalltag eines Informatikers geben. Dazu gehört unter anderem die Kommandozeile. Eine Anforderung der Lernplattform ist deshalb, dass man (optional) neue Aufgaben herunterladen und Lösungsversuche über Befehle in der Kommandozeile testen und abgeben kann. In Code FREAK kann man Aufgaben lediglich über die Oberfläche testen und abgeben.

3.1.3 GitHub Classroom

3.1.3.1 Allgemeines

GitHub Classroom ist ein weiterer Kandidat für den Einsatz an der Hochschule Regensburg. Die Plattform ermöglicht die automatisierte Erstellung von Repositories auf GitHub. Außerdem hilft Classroom dabei, Aufgabenvorlagen und dazugehörige Abgaben einfach zu verwalten und automatisch zu benoten. Dabei enthalten die von GitHub Classroom erstellten Aufgaben-Repositories bereits vorkonfigurierte Zugriffskontrollen. (GitHub Inc., n. d. c)

3.1.3.2 Ablauf für Studierende

Studierende bekommen pro Programmieraufgabe einen Einladungslink. Nach Annahme der Einladung wird pro Studierenden automatisch ein Repository für die jeweilige Aufgabe angelegt. Dieses Repository enthält die Vorlage, welche zum Bearbeiten der Aufgabe benötigt wird.

Sobald der Studierende eine Lösung zur Korrektur abgeben möchte, kann er per Push die Änderungen in das Remote-Repository hochladen. Je nach Konfiguration der Aufgabe starten daraufhin serverseitig ein oder mehrere Tests. Wenn alle Tests bestanden sind, hat der Studierende die Aufgabe erfolgreich abgeschlossen.

3.1.3.3 Architektur

GitHub Classroom ist ein Bildungsservice der Firma GitHub Inc. und ist Stand heute kostenfrei (Arelia Jones, 2020). Die Software basiert auf der Automatisierung von Repositories. Jeder bei GitHub registrierte Nutzer, kann einen sogenannten „Classroom“erstellen und darin Aufgaben auf Basis vorhandener öffentlichen GitHub-Repositories erstellen.

3.1.3.4 Probleme beim Einsatz für die OTH

Es gibt keine Möglichkeit das System von GitHub Classroom auf einen lokalen Git-Server zu replizieren. Aufgrund dessen schafft man sich durch die Verwendung von Classroom eine externe Abhängigkeit an GitHub. Dies kann unter Umständen zu erheblichen Problemen führen, wenn der Dienst beispielsweise nicht erreichbar oder

aufgelöst wird. Ersteres ist durch die Größe und Infrastruktur des Unternehmens nicht (häufig) zu erwarten.

Eine weitere Hürde ist der Bedarf an weiterer Software. GitHub Classroom alleine ist nicht ausreichend, um als vollständige Lernplattform für den Kurs Digital Skills geeignet zu sein. Hier bietet es sich an, eine eigene statische Website mit Anleitungen und Erklärungen zu bauen, welche dann jeweils auf Classroom Einladungslinks verweist. Als Online-Entwicklungsumgebung kann der Dienst Replit verwendet werden. In Replit ist es möglich, ein vorhandenes Git-Repository als Template für eine neue Umgebung zu verwenden. Dieses Template könnte Wrapper-Programme für den git-Workflow enthalten. Der Vorteil daran: Studierende bekommen erste Erfahrungen mit der Kommandozeile, müssen jedoch keine komplexen Git-Kommandos absetzen.

3.2 Finale Architektur Online-Learning Platform

3.2.1 Tutors als Aufgabensammlung

3.2.2 GitHub Classroom als Abgabe- und Bewertungssystem

GitHub Classroom eignet sich, für den Einsatz an der Hochschule, vor allem durch seine Flexibilität und Einfachheit. Durch diese Flexibilität ist es möglich, GitHub Classroom nur als eine austauschbare Komponente des Systems zu sehen. Classroom übernimmt im System die Rolle des Aufgabenservers. Hier werden alle Aufgabenvorlagen, sowie alle Versuche der Studierenden gespeichert und bewertet.

3.2.3 Replit als Online-Entwicklungsumgebung

Der Programmierkurs soll für jeden Teilnehmer ohne komplizierte Installationen durchführbar sein. Aus diesem Grund wurde die Entscheidung gefällt, eine Online Entwicklungsumgebung einzusetzen.

Der Vorteil an Replit ist, dass man dem Studierenden durch ein Vorlagenrepository alle nötigen Konfigurationen und Programme im Vorhinein bereitstellen kann. Sobald ein sogenanntes „Repl“ (Bezeichnung für ein Projekt in Replit) mit der erwähnten Vorlage erstellt wurde, kann der Studierende ohne Installation geräteübergreifend online programmieren (Replit Docs, n. d. b). Der Teilnehmer muss daraufhin lediglich auf den „Run“-Knopf drücken, welcher die später näher erläuterte „OTH-Console“ startet und schließlich alle benötigten Abhängigkeiten bereitstellt.

4 Konfiguration und Implementierung

4.1 Tutors als Aufgabensammlung

4.2 GitHub Classroom

4.2.1 Konfiguration

Für die Einrichtung wurde eine GitHub Organisation erstellt. GitHub Organisationen können von jedem Nutzer GitHub-Nutzer erstellt werden und benötigen lediglich einen Namen und eine Kontakt-E-Mail-Adresse (GitHub Inc., n. d. b). Die Organisation dieses Kurses beherbergt alle Aufgabenvorlagen, die später näher erläuterte Vorlage für Replit, sowie alle Aufgabenrepositories der Studierenden.

4.2.2 Erste Aufgaben

Nach der Erstellung der OTH-Organisation mussten die Aufgabenvorlagen erstellt werden. Aufgabenvorlagen sind in GitHub Classroom normale Repositories, welche in GitHub als Vorlage markiert wurden. Sie beinhalten meist zusätzlich Tests, um den Code darin zu prüfen.

Sobald die Organisation mit Aufgaben gefüllt war, konnte der „Classroom“ für den Kurs angelegt werden. Anfangs ist ein Classroom, wie die Organisation auch, leer. Über die Oberfläche können neue Assignments erstellt werden. Assignments sind Aufgaben, welche dem Studierenden zur Verfügung stehen. Für jedes vorher angelegte Aufgabenrepository wurde ein Assignment erstellt. Bei der Erstellung gibt man verschiedene Konfigurationsparameter an. Dazu gehört die Auswahl, ob eine Aufgabe von Einzelpersonen, oder einer Gruppe bearbeitet werden kann, oder ob die jeweiligen Versuche für alle Studierenden oder nur für die Lehrer sichtbar sind. Ferner gibt es die Möglichkeit eine Deadline, sowie den Starter Code anzugeben. Für den Starter Code wurde in diesem Fall jeweils das Aufgabenrepository ausgewählt. (GitHub Inc., n. d. a)

4.2.3 Tests und Benotung

Im nächsten Schritt legt man die Benotung und das Feedback fest. Das Autograding (die Benotung) geschieht über Kommandos in der Konsole. In unserem Fall beinhal-

tet jedes Aufgabenrepository einen Ordner mit pytest-Tests. Pytest ist eine Code-Test-Bibliothek für Python. Die Assignments wurden so konfiguriert, dass GitHub nach jedem Push zum Repository des Studierenden die pytest-Tests gestartet werden. Wenn alle Tests erfolgreich sind, erhält der Studierende eine vorkonfigurierte Punktzahl. Durch Classroom ist es außerdem möglich, jedem Test eine individuelle Punktzahl zuzuweisen. So kann man dem Schüler neben der erfolgreichen Ausführung beispielsweise noch Bonuspunkte für das Fangen von nicht geplanten Eingaben vergeben. (GitHub Inc., n. d. a)

4.3 Erstellung eines Replit-Starter-Templates

4.3.1 Template-Repository

Projekte in Replit heißen Repls. Diese Repls können auf Basis von GitHub Repositories erzeugt werden. Die Import-Funktionalität ermöglicht das Bereitstellen von Dateien, die die Bearbeitung der Kursaufgaben erleichtern.

Aus diesem Grund wurde ein öffentliches Repository in der vorher erstellten GitHub Organisation angelegt. Dies ist das Template, welches später von den Studierenden als Starter-Vorlage verwendet wird. Replit versteckt Dateien, welche sich in einem Ordner namens `/node_modules` befinden. Normalerweise wird der Ordner automatisch im Kontext mit externen Modulen der JavaScript-Bibliothek Node.js verwendet (npm Inc., n. d.). Das ist auch der Grund weshalb Replit diesen Ordner automatisch versteckt. Dieses Verhalten nutzen wir, um Hilfsprogramme und Konfigurationen zu verstecken.

Studierende werden den ganzen Kurs in einem Repl absolvieren. Jede Programmieraufgabe wird als Ordner im Projekt-Repl abgespeichert werden. Um die Aufgaben herunterladen, prüfen und schließlich abgeben zu können, benötigt man git-Kenntnisse, sowie Erfahrungen mit Test-Frameworks, wie zum Beispiel pytest. Im `/node_modules`-Ordner des Starter-Templates befinden sich diverse Wrapper-Tools, welche dem Studierenden die Arbeit abnehmen und sie dabei unterstützen.

4.3.2 Wrapper-Tools (get, check, submit)

Die Wrapper-Tools `get`, `check` und `submit` sind Python Skripte. Über den Konsolenbefehl `get <PROJEKT-REPOSITORY>` kann der Studierende die Aufgabe in seinen Arbeitsbereich laden. Der Befehl `check <PROJEKT-REPOSITORY>` erlaubt es die Aufgabe auf Fehler zu überprüfen.

Schließlich pusht der Befehl `submit <PROJEKT-REPOSITORY>` den Lösungsversuch in das GitHub Classroom Aufgabenrepository des Kursteilnehmers.

Damit die Skripte Ordnerunabhängig ausgeführt werden können, werden Konsolenaliase benötigt. Konsolenaliase können in einer Bash-Konsole beispielsweise über das Anhängen folgender Zeile an die `.bashrc`-Datei erstellt werden: `alias befehl=ëcho Hallo"`. Bash ist eine Art „Standard-Shell“ unter Linux und wird auch von Replit als Konsole eingesetzt (ubuntu Deutschland e.V., n. d.). Bei jeder neuen Konsolensitzung wird dann der Alias aus der Datei eingelesen und angewendet. Die genannte Datei befindet sich in der Regel im Benutzerverzeichnis, welches außerhalb des Arbeitsbereiches in Replit liegt. Alle Änderungen außerhalb des Arbeitsbereiches werden jedoch von Replit nach jeder Sitzung zurückgesetzt. Um dieses Problem zu beheben, wurde die nun folgende OTH-Console eingeführt.

4.3.3 OTH-Console

Sobald der Student in seinem Repl auf den Run-Knopf drückt, startet die sogenannte OTH-Console in der Konsole. Dies ist eine neue modifizierte Konsoleninstanz, welche alle für die Arbeit benötigten Konfigurationen enthält.

Sobald der Run-Knopf gedrückt wird, startet das Einrichtungsskript `setup.py`, welches die benötigten Dateien in das Benutzerverzeichnis schreibt.

Zuerst wird eine Konfigurationsdatei für Bash angelegt. In diese werden alle benötigten Aliase (`get`, `submit`, `check` und `github`) geschrieben. Außerdem wird GitHub, falls noch nicht vorhanden, zu den sogenannten „Known Hosts“ im SSH-Ordner hinzugefügt. Der Vorteil daran ist, dass der Student bei der ersten Verbindung mit GitHub (bspw. durch den `get`-Befehl) keine Authentizitätsprüfung bestätigen muss (ssh.com, n. d.). Als letztes wird in der Konfigurationsdatei noch das Aussehen des Bash Promptes festgelegt.

Im nächsten Schritt wird die passwortlose Authentifizierung mit GitHub eingerichtet. Hierzu benötigt man ein SSH-Schlüssle-Paar, welches automatisch, falls nicht vorhanden, durch das Einrichtungsskript erzeugt wird. Nach der Erzeugung wird es neben dem SSH-Ordner auch in die Repl-Nutzer-Datenbank geschrieben. Die Datebank ist ein simpler Key-Value-Speicher, welcher jeweils pro Replit-Projekt existiert (Replit Docs, n. d. a). Sobald der Studierende Replit neustartet und das Benutzerverzeichnis gelöscht wurde, holt sich das Einrichtungsskript die SSH-Keys aus der Datenbank und schreibt sie wieder zurück in die jeweiligen Dateien. Dasselbe Verfahren wird für die Konfiguration von Git angewandt. Git benötigt, um

Änderungen zu pushen, einen Namen mit zugehöriger E-Mail Adresse (Software Freedom Conservancy, n. d.). Diese Daten werden zusammen mit den SSH-Keys in der Replit-Datenbank gespeichert.

Nach der Ausführung des Einrichtungsskripts, wird eine neue Bash-Konsolen-Instanz, mit der gerade angelegten Konfigurationsdatei als Parameter, gestartet.

4.3.4 SSH-Keys

Um das vorher generierte SSH-Schlüsselpaar für die Authentifizierung gegen GitHub zu verwenden, muss der öffentliche Schlüssel noch zu dem GitHub Profil des Studierenden hinzugefügt werden. Hierfür enthält das Starter-Template ein weiteres Programm, welches mit dem Befehl `github` in der OTH-Console ausgeführt werden kann. Dieses weitere Python Programm lädt den im SSH-Ordner gespeicherten öffentlichen Schlüssel und gibt ihn zusammen mit einem Link zum Hinzufügen von SSH-Keys in GitHub aus.

Desweiteren überprüft das Programm, ob Git bereits konfiguriert ist. Sind die benötigten Werte nicht in der Replit-Datenbank vorhanden, frägt das Programm den Nutzer nach dem Name und der studentischen E-Mail Adresse, welche schließlich beide in der Datenbank hinterlegt werden. Außerdem werden die Werte, wie beim Einrichtungsskript auch, in eine Git-Konfigurationsdatei geschrieben.

5 Studie: Test an fachfremden Studierenden

6 Zusammenfassung und Ausblick

Abbildungsverzeichnis

1 CS50 Architektur 6

Literaturverzeichnis

Arelia Jones. (2020). *Set up your digital classroom with GitHub Classroom*. Verfügbar 28. Januar 2022 unter https://classroom.github.com

GitHub Inc. (n. d. a). *Create an individual assignment*. Verfügbar 7. Februar 2022 unter https://docs.github.com/en/education/manage-coursework-with-github-classroom/teach-with-github-classroom/create-an-individual-assignment

GitHub Inc. (n. d. b). *Creating a new organization from scratch*. Verfügbar 7. Februar 2022 unter https://docs.github.com/en/organizations/collaborating-with-groups-in-organizations/creating-a-new-organization-from-scratch

GitHub Inc. (n. d. c). *GitHub Classroom*. Verfügbar 28. Januar 2022 unter https://classroom.github.com

Harvard University. (n. d. a). *CS50 GitHub*. Verfügbar 28. Januar 2022 unter https://github.com/cs50

Harvard University. (n. d. b). *CS50: Introduction to Computer Science*. Verfügbar 28. Januar 2022 unter https://pll.harvard.edu/course/cs50-introduction-computer-science

Harvard University. (n. d. c). *CS50's Introduction to Computer Science*. Verfügbar 28. Januar 2022 unter https://www.edx.org/course/introduction-computer-science-harvardx-cs50x

Harvard University. (n. d. d). *Online - CS50 Docs*. Verfügbar 28. Januar 2022 unter https://cs50.readthedocs.io/ide/online/

Harvard University. (n. d. e). *submit50 - CS50 Docs*. Verfügbar 28. Januar 2022 unter https://cs50.readthedocs.io/submit50

Kiel University of Applied Sciences. (n. d. a). *Code FREAK Documentation :: Code FREAK Docs*. Verfügbar 28. Januar 2022 unter https://docs.codefreak.org/codefreak/index.html

Kiel University of Applied Sciences. (n. d. b). *Installation Guide :: Code FREAK Docs*. Verfügbar 28. Januar 2022 unter https://docs.codefreak.org/codefreak/for-admins/installation.html#_dedicated_docker_host

Kiel University of Applied Sciences. (2019-2020). *Code FREAK | Code Feedback, Review & Evaluation Kit*. Verfügbar 28. Januar 2022 unter https://codefreak.org/#about

LDAP. (n. d.). *LDAP.com - Lightweight Directory Access Protocol*. Verfügbar 28. Januar 2022 unter https://ldap.com

moodle.de. (n. d.). *Lernerfolg mit Moodle*. Verfügbar 28. Januar 2022 unter https://moodle.de

Mouat, A. (2015). *Using Docker: Developing and Deploying Software with Containers*. O'Reilly Media, Inc.

npm Inc. (n. d.). *folders - Folder Structures Used by npm*. Verfügbar 7. Februar 2022 unter https://docs.npmjs.com/cli/v7/configuring-npm/folders

Red Hat. (2018). *Was ist ein Linux-Container?* Verfügbar 31. Januar 2022 unter https://www.redhat.com/de/topics/containers/whats-a-linux-container

Replit Docs. (n. d. a). *Database FAQ*. Verfügbar 7. Februar 2022 unter https://docs.replit.com/hosting/database-faq

Replit Docs. (n. d. b). *Replit and GitHub: Using and contributing to open-source projects*. Verfügbar 7. Februar 2022 unter https://docs.replit.com/tutorials/06-github-and-run-button

Software Freedom Conservancy. (n. d.). *1.6 Getting Started - First-Time Git Setup*. Verfügbar 7. Februar 2022 unter https://git-scm.com/book/en/v2/Getting-Started-First-Time-Git-Setup

ssh.com. (n. d.). *SSH Host Key - What, Why, How*. Verfügbar 7. Februar 2022 unter https://www.ssh.com/academy/ssh/host-key

ubuntu Deutschland e.V. (n. d.). *Bash*. Verfügbar 7. Februar 2022 unter https://wiki.ubuntuusers.de/Bash/