Praktikum Software-Engineering 1

Sommersemester 2025 – Blatt 2

Prof. Dr. Stefan Sarstedt <stefan.sarstedt@haw-hamburg.de>
Enrico Christophers <enrico.christophers@haw-hamburg.de>

Bearbeitungshinweise

- Es gelten dieselben Hinweise wie für Blatt 1.
- **Achtet auf Rechtschreibung und Grammatik!**
- Abgabe der Lösungen in einem einzelnen PDF per Mail an mich und an enrico.christophers@haw-hamburg.de

Aufgabe 1: Glossar

Was ist spezielles Vokabular in Eurem Projekt, das man missverstehen kann? Erstellt ein Glossar, in dem Ihr diese Begriffe definiert, damit es im Projekt nicht zu Missverständnissen zwischen Euch und dem Kunden (oder untereinander) kommt.

Aufgabe 2: INVEST / Überarbeitung der User Stories

Untersucht Eure User Stories für das BUS im Hinblick auf die INVEST-Kriterien und verbessert die Stories entsprechend, falls sie diesem Kriterien nicht genügen!

Aufgabe 3: ER-Modell

Erstellt ein ER-Datenmodell (siehe Vorlesung Datenbanken) für Euer System (bestehend aus BUS und BMS!). Nutzt hierzu z.B. http://draw.io/.

Aufgabe 4: Setup des Beispielprojekts

a) Klont das Beispielprojekt aus der Vorlesung:
`git clone git@git.haw-hamburg.de:lehre-sose-25/srs/se1/se1lab.git` (SSH) oder
`git clone https://git.haw-hamburg.de/lehre-sose-25/srs/se1/se1lab.git` (HTTPS)
Um Zugriff auf das Projekt zu bekommen, müsst ihr Euch **mindestens einmal** auf git.haw-hamburg.de eingeloggt haben. Ihr könnt dann dem Projekt beitreten.
p.s. es macht Sinn, einen SSH-Key in gitlab einzurichten. Anleitung siehe dort.

b) **Löscht den lokalen .git-Ordner des geklonten Projekts**. Damit entfernt ihr die Verbindung zu meinem gitlab-Projekt.

c) Führt die Installations-/Konfigurationsschritte in der README.md des Projekts durch. Wir empfehlen, JetBrains IntelliJ Ultimate in der neuesten Version zu verwenden (wir können keinen bzw. nur wenig Support für andere IDEs geben).

d) Führt die Übungen Lab 2/3/4 aus der entsprechenden Vorlesung (siehe Folien dort) durch.

e) Erzeugt ein neues Projekt in Eurem Namespace auf git.haw-hamburg.de: New project→Create blank project→dort den Haken bei „Initialize repository with a

README“ **entfernen**. Nennt Euer Projekt ebenfalls „se1lab“. Anleitung zum Pushen Eures (neuen) Projekts dann unter dem Abschnitt „**Push an existing folder**“.

f) Gebt uns Euer Projekt auf Gitlab mit Berechtigung "Maintainer" frei. Setzt Eure Klarnamen für Eure Profile (rechts oben Settings).

Aufgabe 5: Implementierung und Test

Führt folgende Änderungen/Ergänzungen an Eurem Code der vorherigen Aufgabe durch:

a) Benennt die Klasse Customer in „Lecturer“ um. Nutzt hierzu die Refactoring-Funktionalität Eurer IDE (Refactor→Rename in IntelliJ)!

b) Erstellt einen fachlichen Datentyp „Address“ mit den Attributen Street, Number, ZIP und City. Ergänzen Sie die Klasse Lecturer um die Adresse. Orientiere Dich an der Einbettung von Phonenumber in Lecturer. Achte auf korrekte Annotationen!
Hinweis: verwendet für die Attribute der Adresse nur die Wrapper-Klassen der Datentypen (also z.B. „Integer“ anstelle von „int“). Ansonsten kommt eine Fehlermeldung.

c) Erstelle eine Entität „Participant“ mit den Attributen Id (Typ Long) und Name (Typ String). Die Klasse Course soll eine Liste von „Participant“ enthalten (also eine Liste von Teilnehmern des Kurses). Orientiere Dich an der Verbindung Lecturer→Course. Achte auf korrekte Annotationen! Diese sind teilweise anders als in b)!

d) Implementiere folgende Funktionen in der Klasse „LecturerService“ (vormals CustomerService!):
   - createParticipant(String name) → legt einen Participant an und gibt die neue Id zurück. Soll einen Fehler liefern (eigene Exception schreiben! Achte auf einen guten Namen dafür!) falls es bereits einen Participant mit demselben Namen gibt.
   - enrollParticipant(Long courseId, Long participantId) → Fügt einen Partitipant einem Kurs hinzu; mit entsprechender Fehlerbehandlung, z.B. falls es den Course oder den Participant unter dieser Id nicht gibt.

e) Schreibe angemessene Testfälle für die Funktionen aus d). Teste sowohl Erfolgs-, als auch Misserfolgsszenarien (Kurs nicht vorhanden, Participant nicht vorhanden, etc.)!

f) Stellt sicher, dass die Tests lokal erfolgreich sind ("./gradlew clean build").

g) Pusht die Änderungen in Euer Git-Repository. Achtet auf "gute" Commit-Messages!

Viel Spaß!

Stefan & Enrico