Praktikum „Software Engineering 1“

Aufgabenblatt 1

Wintersemester 2021/2022

Hinweise

Betreuer

- Arne Busch <arne.busch@haw-hamburg.de>
- Sven Berding <sven.berding@haw-hamburg.de>

Bearbeitungshinweise

- **Teamarbeit**: Die Bearbeitung erfolgt in festen 2er-Teams. Jeder muss sich adäquat beteiligen.
- **Abgabe**: Die Aufgabenlösungen müssen bis zum Praktikumstermin vollständig fertiggestellt und abgegeben worden sein. Die Abgabe von Lösungen in Dateiform erfolgt durch ein Hochladen in das Git-Repository des eigenen Praktikumsprojekts. Nicht-Code-Abgaben (PDFs etc.) werden in folgendem Ordner abgelegt:
  - *<Projekt-Repository>/aufgabenblaetter/<Aufgabenblattnummer>*
  - Beispiel: *https://git.haw-hamburg.de/se1-lab_meyer_schulze/.../aufgabenblaetter/1*

  Der Dateiname jeder dort abgelegten Datei muss gemäß folgendem Schema aufgebaut sein:
  - *aufgabe_<Aufgabennummer>*
  - Beispiel: *aufgabe_1.2.pdf*
- **Abnahme**: Im Praktikumstermin erfolgt eine Prüfung und Abnahme der Aufgabenlösungen durch Ihren Betreuer. Bei unzulänglichen Lösungen wird eine Nacharbeitung gefordert.
- **Nacharbeiten**: Ihr Betreuer dokumentiert die noch einmal zu überarbeitenden Lösungen mittels einer Checkliste in einem Issue für Ihr GitLab-Projekt (*Projektseite > Issues > Aufgabenblatt <Nummer> - Nacharbeiten).* Nach der Abarbeitung eines Punktes haken Sie diesen ab. Nachdem alle Punkte abgehakt sind, benachrichtigen Sie Ihren Betreuer darüber (per Mail). Waren die Nacharbeiten zufriedenstellend, markiert Ihr Betreuer den GitLab-Issue als "Closed" und wertet das Aufgabenblatt als "bestanden"; falls nicht, entfernt er den Haken bei den nochmals zu überarbeitenden Punkten und informiert Sie über die Notwendigkeit weiterer Nacharbeiten. Ab hier wiederholen sich die Schritte im Nacharbeitsprozess (s. o.).
- **Feedback**: Feedback zu den Aufgabenlösungen wird nur in der Abnahme gegeben. Für die finale Abgabe wird daher auch keine Korrektur ausgegeben, da die (ggf. noch einmal überarbeiteten) endgültigen Lösungen bereits besprochen wurden und i. d. R. ok sein sollten.
- **Teilnahmebedingungen**: Eine erfolgreiche Abnahme sowie die fristgerechte Abgabe bzw. Nacharbeitung der Lösungen sind Voraussetzung für die weitere Teilnahme am Praktikum.
- **Bonusaufgaben**: Bonuspunkte führen zu Zusatzpunkten in der Prüfung (alle Bonuspunkte auf allen Aufgabenblättern ≙ 10 % der Prüfungspunkte; anteilige Bonuspunkte entsprechend).
- **Fristen**: Folgende Deadlines gelten für die Abnahme und Abgabe der Nacharbeiten:
  - Gruppe 1: Abnahme: 03.11.2021, Uhrzeit: Slots; Abgabe: 05.11.2021, 12:00 Uhr
  - Gruppe 2: Abnahme: 10.11.2021, Uhrzeit: Slots; Abgabe: 12.11.2021, 12:00 Uhr
  - Gruppe 3: Abnahme: 17.11.2021, Uhrzeit: Slots; Abgabe: 19.11.2021, 12:00 Uhr

Aufgabe 1.1: Setup Guide

Lesen Sie den Setup Guide für die Vorlesung. Er enthält Hintergrundinfos und Anleitungen für die Einrichtung von Tools und Projekten, die im weiteren Verlauf benötigt werden.

Hinweis: Der Setup Guide findet sich hier: Vorlesungs-Team > Kanal „Allgemein“ > Dateien > Skript

Abgabe: Keine (Prüfung während der Abnahme)

Aufgabe 1.2: Entwicklungs-Tools

Richten Sie auf ihrem Rechner die folgenden Tools ein (gemäß den Anleitungen im Setup Guide):

- Git
- Git Client (optional)
- JDK (JDK 8 bzw. Version 1.8.x)
- IDE (Eclipse oder IntelliJ)

Hinweis: Die Installation und Verwendung der vorgegebenen JDK-Version ist essenziell für die korrekte Lauffähigkeit der Projekte, da eine vollständige Kompatibilität mit den verwendeten Frameworks sonst nicht sichergestellt werden kann.

Abgabe: Keine (Prüfung während der Abnahme)

Aufgabe 1.3: Referenzprojekt

a)

Richten Sie das Referenzprojekt auf ihrem Rechner ein. Führen Sie dabei die folgenden Schritte aus:

- Referenzprojekt herunterladen (klonen)
- In Haupt-Branch des Semesters wechseln (*ws2021/master*)
- Referenzprojekt in IDE importieren
- Referenzprojekt bauen und testen (via „gradlew build“)
- Server-Anwendung starten (Main-Klasse in IDE ausführen)
- Web Client testen:
    - Seite im Webbrowser öffnen (*<Projektverzeichnis>/src/main/webapp/ping.html*)
    - Button „Ping Server“ klicken
    - Bei „Last ping response“ sollte nun etwas angezeigt werden

Hinweis: Details zum Referenzprojekt finden sich im Setup Guide.

Hinweis: Aktuell befindet sich das Referenzprojekt noch im Status „Proof of Concept“, womit zunächst das technische Grundgerüst aufgebaut und die Umsetzbarkeit überprüft wird. Daher besitzt die Anwendung bisher nur eine Funktionalität, um den Server vom Client aus anzupingen.

Abgabe: Keine (Prüfung während der Abnahme)

b)

Machen Sie sich mit den wesentlichen Konzepten des Spring Frameworks vertraut (Application Context, @Component, @Autowired).

Informieren Sie sich ebenfalls über das Datenformat JSON, welches für den Datenaustausch zwischen Client und Server verwendet wird.

Studieren Sie den Code der Server-Anwendung (*<Projektverzeichnis>/src/main/java*). Schauen Sie sich auch die Testklassen für die Server-Anwendung an (*<Projektverzeichnis>/src/test/java*).

Vollziehen Sie die einzelnen Entwicklungsschritte nach, indem Sie die Git-Historie des Projekts durchgehen und sich die Diffs jedes Commits im folgenden Zeitraum anschauen:

- Start: Commit-ID 1d5a6774, 10.04.2021, Added directory for test sources
- Ende: Commit-ID 196a29e8, 11.04.2021, Added basic web client for ping requests

Hinweis: Die Git-Historie von Projekten ist komfortabel in GitLab einsehbar. Hier ist die Historie in GitLab zu finden: *Projektseite > Repository > Commits*

Hinweis: Mit den Details der Serveranbindung via REST (@RestController, @GetMapping usw.) brauchen Sie sich hier noch nicht zu beschäftigen, da dies später in der Vorlesung behandelt wird.

Hinweis: Der Code des Web Clients (*<Projektverzeichnis>/src/main/webapp*) braucht nicht studiert zu werden, da die Entwicklung von Weboberflächen und GUI-Programmierung nicht Inhalt von SE1 sind. Für Interessierte kann sich allerdings ein Blick in den Code lohnen, um die Funktion der Gesamtanwendung besser zu verstehen.

Abgabe: Keine (Prüfung während der Abnahme)

c)

Stellen Sie einen Client-Server-Roundtrip grafisch dar, d. h. zeichnen Sie schematisch auf, was passiert, wenn der Benutzer im Web Client auf den Button „Ping Server“ klickt, und zeigen Sie auf, was noch alles geschieht, bis der Roundtrip mit der Ausgabe des Rückgabewerts des Server-Aufrufs im Web Client endet. Geben Sie dabei in jedem Schritt an, in welchem Format die verarbeiteten Daten jeweils vorliegen bzw. was jeweils für Daten übergeben und welche zurückgegeben werden (ggf. mit Beispielen).

Hinweis: Beschränken Sie sich auf die Anwendungsteile, deren Code selbst geschrieben ist, d. h. forschen Sie beispielsweise nicht in den Tiefen von Framework-Code nach, wo genau eine Umwandlung von JSON-Daten in Java-Objekte stattfindet. Der gesamte Roundtrip sollte nicht mehr als 5-6 Schritte aufzeigen.

Hinweis: Wesentliche Daten werden sowohl in der Konsole der IDE (Server) als auch in der Konsole des Webbrowsers (Client) geloggt. Die Webbrowser-Konsole lässt sich über die Taste F12 öffnen.

Hinweis: Zum schrittweisen Nachvollziehen eines Client-Server-Roundtrips auf der Server-Seite kann man den Debugger der IDE nutzen. Hier am besten einen Breakpoint in der Klasse *PingService* setzen.

Abgabe: Grafische Darstellung eines Client-Server-Roundtrips als PDF

Aufgabe 1.4: Eigenes Praktikumsprojekt

a)

Wählen Sie ein zum Motto passendes Thema für Ihre Praktikumsanwendung, d. h. die Art bzw. den Zweck der Software, die Sie während des Praktikums entwickeln wollen. Optional können Sie Ihrer Anwendung zusätzlich einen Namen/Arbeitstitel geben.

Hinweis: Wichtige Details zu Thema, Motto u. v. m. finden sich im Skript „Organisatorisches“.

Abgabe: Keine (Prüfung während der Abnahme)

b)

Erstellen Sie das Grundgerüst des Projekts für Ihr Praktikumsteam. Führen Sie dabei die folgenden Schritte aus:

- Gruppe für Projektteam erstellen (in GitLab)
- Basisprojekt kopieren (forken)
- Eigenes Projekt konfigurieren
- Teampartner und alle Betreuer (s. o.) mit Rolle „Maintainer“ zum eigenen Projekt hinzufügen
- Pipeline des eigenen Projekts starten (Button „Run Pipeline“ auf Pipeline-Seite in GitLab)
- Eigenes Projekt herunterladen (klonen)
- Eigenes Projekt in IDE importieren
- Eigenes Projekt bauen und testen (via „gradlew build“)
- Eigene Anwendung starten/schließen (via „gradlew bootRun“ und STRG+C für Terminierung)
- Ordner für Lösungen der Aufgabenblätter im eigenen Projekt anlegen:
    - Hauptordner: *<Projektverzeichnis>/aufgabenblaetter*
    - Aufgabenblatt 1: *<Projektverzeichnis>/aufgabenblaetter/1*

Hinweis: Details zur Einrichtung des eigenen Praktikumsprojekts finden sich im Setup Guide.

Hinweis: In GitLab können für Projekte Build Pipelines aufgesetzt werden (auch als „Continuous Integration Pipelines“ oder „Continuous Delivery Pipelines“ bezeichnet, kurz „CI/CD“). Mit einer Pipeline wird automatisiert nach jedem Push ins Projekt-Repository ein Build & Test durchgeführt. Falls ein Pipeline-Durchlauf fehlschlägt, d. h. wenn beispielsweise nicht kompilierbarer Code gepusht wurde oder wenn Unit Tests fehlschlagen, bekommen alle Projektmitglieder eine E-Mail mit dem Hinweis, den Fehler in der Code Base zu fixen. Pipelines tragen damit wesentlich zur Codequalität bei. Hier sind die Pipelines eines Projekts in GitLab zu finden: *Projektseite > CI/CD > Pipelines*

Hinweis: Die Pipeline eines Projekts sollte nie (NIE!) längerfristig fehlschlagen, d. h. ein kaputtes Projekt sollte stets unverzüglich gefixt werden. Der Aufwand zum Fixen eines Projekts, auf dem trotz fehlschlagender Pipeline einfach weiter entwickelt wurde, wächst exponentiell an, da sich irgendwann Fehler ansammeln und Fehlerursachen nicht mehr klar den Änderungen aus einem bestimmten Commit/Push zugeordnet werden können. Daher sollte man während der Entwicklung auch regelmäßige, eher kleinere Commits machen und die lokalen Commits in nicht allzu großen Abständen ins Repository pushen.

Abgabe: Einladung zum Projektbeitritt an alle Betreuer

Aufgabe 1.5: Persona

a)

Definieren Sie eine Proto-Persona aus einer Benutzergruppe Ihrer Praktikumsanwendung. Beschreiben Sie die Persona so konkret wie möglich, z. B. mit folgenden Eigenschaften:

- Persönliche Daten
- Biografie
- Erfahrung
- Charakter
- Einstellungen
- Ziele

Abgabe: Proto-Persona als PDF

b)

Schildern Sie kurz anhand einer bestimmten Eigenschaft/Besonderheit der Persona, welchen Mehrwert diese Eigenschaft gegenüber einer abstrakten Benutzergruppe darstellt, z. B. welchen Einfluss dies auf die Anforderungen haben könnte.

Abgabe: Keine (Prüfung während der Abnahme)

Aufgabe 1.6: Value Proposition Canvas

Erstellen Sie einen Value Proposition Canvas für die Benutzergruppe Ihrer Proto-Persona, um mit dessen Hilfe mögliche Services (Features) abzuleiten, welche durch Ihre Praktikumsanwendung realisiert werden könnten. Tragen Sie in jedem Canvas-Bereich jeweils mindestens drei Items ein.

Hinweis: Beachten Sie die Regeln für gute Canvases. Achten Sie beispielsweise in den Pain Relievers und Gain Creators auf konkrete Bezüge zu den aufgeführten Pains bzw. Gains und definieren Sie die Gains nicht lediglich als Negationen der Pains. Abgabe: Value Proposition Canvas als PDF

Aufgabe 1.7: User Stories

Erstellen Sie User Stories für Ihre Praktikumsanwendung. Definieren Sie mindestens 3 komplexere bzw. umfangreichere User Stories als Epics. Splitten Sie die Epics anschließend jeweils sinnvoll in mehrere einfache User Stories auf.

Insgesamt sollen mindestens 10 einfache User Stories vorhanden sein. Erstellen Sie hierzu ggf. weitere einfache User Stories (ohne zgh. Epic). Formulieren Sie für jede einfache User Story mindestens einen Akzeptanztest.

Jede User Story sollte die folgenden Attribute haben:

- Nummer
- Titel

- Beschreibung
- Akzeptanztests (nur für einfache User Stories)

Hinweis: Optional können für einfache User Stories zusätzliche Attribute (Geschäftswert, Entwicklungsaufwand usw.) angegeben werden. Diese helfen eventuell bei der Priorisierung und Einteilung in Releases (s. a. Aufgabe zur User Story Map).

Abgabe: User Stories als PDF

Ab hier kommen Zusatzaufgaben, die freiwillig bearbeitet werden können, um die zgh. Vorlesungsinhalte praktisch anzuwenden. Eine Nichtbearbeitung hat keine Konsequenzen für die Abnahme oder das Bestehen des Aufgabenblatts. Ebenso brauchen eventuelle Unzulänglichkeiten nicht nachgearbeitet zu werden.

Mit diesen Aufgaben können Sie Bonuspunkte sammeln. Die Höhe der erreichten Bonuspunkte wird von Ihrem Betreuer in der Abnahme festgelegt. Für die Bestimmung der erreichten Bonuspunkte ist die in der Abnahme vorgelegte Lösung maßgeblich, d. h. durch ein späteres Nacharbeiten lässt sich die Anzahl der erreichten Bonuspunkte nicht mehr erhöhen.

Aufgabe 1.8: User Story Map [Bonusaufgabe #1]

Erreichbare Bonuspunkte für diese Aufgabe: 2

Erstellen Sie eine User Story Map mit Ihren zuvor erstellten User Stories. Es soll für jedes Epic ein User Task definiert werden, unter das sich die zgh. einfachen User Stories einteilen lassen. Teilen Sie die einfachen User Stories anschließend auf 3 Releases auf (sinnvoll chronologisch angeordnet und möglichst gleichmäßig nach grob abgeschätztem Entwicklungsaufwand).

Hinweis: Optional können pro Release mehrere Iterationen (Slices) definiert werden, um eine detailliertere Unterteilung der User Stories zu erreichen und diese somit nach Priorität/Abhängigkeit bzw. zeitlicher Planung innerhalb eines Releases zu gruppieren.

Hinweis: Optional können zusätzliche User Activities definiert werden, um die User Tasks noch weiter zu strukturieren und damit inhaltlich zusammengehörige User Tasks zusammenzufassen.

Abgabe: User Story Map als PDF

Aufgabe 1.9: Kanban Board [Bonusaufgabe #2]

Erreichbare Bonuspunkte für diese Aufgabe: 2

a)

Erstellen Sie ein Kanban Board für Ihre Praktikumsanwendung. Das Board soll den gesamten Lebenszyklus für einzelne Features abbilden. Definieren Sie hierfür passende Spalten.

Befüllen Sie Ihr Board anschließend mit Einträgen für alle Ihre zuvor erstellten User Stories (nur einfache User Stories, keine Epics). Ordnen Sie die Stories im Board gemäß ihren Prioritäten sowie dem aktuellen Stand der Entwicklung.

Hinweis: Sie können für diese Aufgabe beispielsweise das Feature „Issue Boards“ in GitLab (*Projektseite > Issues > Boards*) oder Trello (https://trello.com) nutzen.

Abgabe: Keine (Prüfung während der Abnahme)

b)

Machen Sie das Board für alle Betreuer zugreifbar, damit sie das Board einsehen können.

Hinweis: Wie Sie Ihr Board für die Betreuer zugreifbar machen, hängt von der Art des Boards ab. In GitLab ist der Zugriff i. d. R. automatisch gegeben, sofern man einen entsprechenden Zugriff auf das Projekt hat. Bei anderen Anbietern muss ein Board ggf. manuell mit anderen Personen geteilt werden (z. B. über ein Einladen- oder Teilen-Feature).

Abgabe: Board-Zugang für die Betreuer