Praktikum Software-Engineering 1

Sommersemester 2025 – Blatt 4

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

Aufgabe 1: Implementierung von Businesslogik

Implementiert in Eurer SpringBoot-Applikation aus Aufgabe 3 (Blatt 3) die folgende Businesslogik (in entsprechenden Seviceklassen):

1. Anlegen von Benutzern.
2. Buchen einer konkreten Fährfahrt (mit Start/Ziel und Kabine).

Alle weiteren (Test-)Daten legt ihr direkt im Code mit entsprechenden Repositories an (Testdaten wie z.B. Häfen, Schiffe, Route, etc., die Ihr benötigt um die obigen Funktionen umzusetzen).

Aufgabe 2: Implementierung einer REST-Fassaden-API

a) Startet die **ursprüngliche se1lab-Applikation (ohne Eure Änderungen)** mittels ./gradlew bootRun (macOS, Linux, Windows Powershell PS – unter Windows Command (cmd) mittels gradlew.bat bootRun) – gradlew bleibt bei ca. 80% stehen, was anzeigt, dass die Applikation nun läuft. Testet die vorhandenen REST-Endpunkte (für GET, POST, PUT, DELETE) mittels Postman (https://www.postman.com, grafisch), httpie (https://httpie.io, grafisch und Kommandozeile) oder curl (https://curl.se, Kommandozeile). Legt einige Daten an, ändert und löscht sie wieder. Lasst Felder weg und schaut, was passiert. Versucht es mit „falschen“ Datenformaten (invalides JSON etc.) Was passiert? Ist die API „robust“? Findet Ihr einen „nicht robusten“ Endpunkt?

b) Erstellt eine REST-Fassade für Eure Applikation **aus Aufgabe 1 oben**. Überlegt Euch sinnvolle REST-Endpunkte, um Eure Fachlogik (die Systemoperationen aus Eurem Use Case) zu einem (zunächst) imaginären Frontend hin anzubieten. Die RestController-Methoden **delegieren** dann an Eure **Service**- bzw. **Repository**-Klassen.

c) Testet Eure Endpunkte manuell mittels Postman, httpie oder curl.

d) Schreibt automatisierte REST-API-UnitTests mittels des RestAssured-Frameworks („given..when...then“). Arbeitet Euch mit Hilfe der Webseite https://rest-assured.io und des gegebenen Beispielcodes in das Framework ein.

Aufgabe 3: Vibe-Coding mit AI 🦄

In dieser Aufgabe versucht ihr, eine komplette Applikation ohne (viel) zu Programmieren zu erstellen. 😊🎉

1. Besorgt Euch einen GitHub Education-Account bzw. -Status: https://github.com/education/students. Hiermit könnt ihr kostenlos das AI-Plugin GitHub Copilot (Chat & Agent) für Visual Studio Code und JetBrains IntelliJ benutzen. Falls Ihr einen Google-Account habt, könnt ihr auch Google Gemini Code Assist (ebenfalls als Plugins für Visual Studio Code und JetBrains IntelliJ verfügbar) verwenden.

Claude Code (https://www.anthropic.com/claude-code) ist auch super, kostet allerdings!

2. Formuliert entsprechende Prompts, die Euch eine BUS-Applikation aus Euren Anforderungen erstellen! (Startet bei Null!)
   Gebt dabei an:
   - dass ihr eine Java Spring Boot Applikation generieren möchtet und Gradle verwendet werden soll
   - ein Frontend in React erstellt werden soll
   - eine gitlab-CI-Pipeline Konfiguration erstellt werden soll
   - Tests generiert werden sollen
3. Im Praktikum lassen wir die anderen Gruppen Eure Anwendung ausprobieren und bewerten!

Hinweise:

- Merkt Euch die Prompts und Euren „Vibe-Coding-Entwicklungsprozess (VCE)“ ™. Dokumentiert, wie ihr vorgegangen seid, welche Probleme ihr hattet und wie und ob ihr sie lösen konntet! Musstet ihr oft manuell eingreifen und Code schreiben oder debuggen? Macht auch Screenshots von Zwischenergebnissen!
- Eventuell müsst ihr eure Anforderungen sehr viel genauer formulieren, damit die KI das tut, was ihr Euch vorstellt.🙂
- Beschreibt der KI auch das Datenmodell und andere Dinge aus Eurer Spezifikation, die helfen können (Wireframes, Use Cases, etc.)
- „Sh*t in – Sh*t out“!
- Startet mit wenigen Anforderungen und arbeitet Euch iterativ vor.
- Versucht den Code zu verstehen, der generiert wird. Werden Libraries eingebaut, mit denen wir uns noch nicht beschäftigt haben? Was machen sie? Lasst Euch von der KI den Code erklären!
- Lasst die KI bei Fehlern auch die Bugfixes machen.
- Wo habt ihr Probleme und wo liegen für Euch die Grenzen? War es hilfreich?
- Seid ihr zufrieden mit der Codequalität? Wie sieht es mit der Wartbarkeit aus? Ist eine Architektur klar erkennbar? Wurden Entwurfsprinzipien beachtet? Ist die Architektur konsistent? Stellt diese Fragen auch der KI bzw. gebt bei der Generierung auch hierzu konkrete Anweisungen!

Viel Spaß!

Stefan & Enrico