

Reflexionsbericht: Frontend-Entwicklung und Prozessintegration im Projekt „StudIQ“

Im Rahmen des Moduls Software Engineering verstärkte ich das Projektteam ab der zweiten Projekthälfte als Frontend-Entwickler. Der Einstieg in ein bereits laufendes Projekt nach dem Scrum-Framework stellte eine besondere Herausforderung dar, da sowohl die technische Architektur als auch die organisatorischen Prozesse und die Teamdynamik bereits etabliert waren. Meine Aufgabe bestand darin, mich innerhalb kürzester Zeit in die bestehende Codebase einzuarbeiten, die Vision des Produkts zu verinnerlichen und produktive Beiträge zu leisten, um die finalen Meilensteine des MVP zu sichern.

1. Onboarding und Integration in bestehende Strukturen

Ein wesentlicher Teil meiner Einzelleistung bestand in der effizienten Einarbeitung. Um den Projekterfolg nicht durch lange Anlaufzeiten zu verzögern, wendete ich strukturierte Methoden des Reverse Engineering an. Ich analysierte die vorhandene Frontend-Architektur sowie die bereits implementierte Schichtenarchitektur (Django/Vue.js), um sicherzustellen, dass meine Beiträge technologisch konsistent blieben.

Da die initiale Systemarchitektur zu Beginn nur teilweise dokumentiert war, nutzte ich die Weekly Meetings intensiv, um Wissenslücken zu schließen und mich mit den Backend-Schnittstellen vertraut zu machen. Dass ich bereits nach wenigen Tagen voll einsatzfähig war und erste User Stories im Sprint-Backlog übernehmen konnte, belegt mein tiefes Verständnis für modulare Softwarearchitekturen und agile Prozesse.

2. Relevanter Beitrag zum Projekterfolg

Trotz des späten Einstiegs habe ich zentrale, nutzerzentrierte Features verantwortet, die maßgeblich zur Fertigstellung des MVP beigetragen haben:

Studiengang-Overview & Modulverlinkung: Dies war eines der komplexesten UI-Features. Ich implementierte die Logik, um Studiengangsdaten – welche über automatisierte Skripte im Backend bereitgestellt wurden – dynamisch darzustellen und die zugehörigen Module zu verlinken. Dies bildet das funktionale Herzstück der App.

UI-Design & Konsistenz: Da in der ersten Projekthälfte noch kein einheitliches Design-System etabliert war, wiesen die Oberflächen stilistische Inkonsistenzen auf. Ich übernahm die Aufgabe, die UI zu harmonisieren. Ich stellte sicher, dass Farben, Abstände und Komponenten über alle von mir entwickelten Module (Settings, Freundesliste, Overview) hinweg einem einheitlichen Standard folgten und trug so maßgeblich zur professionellen Außenwirkung der App bei.

Settings & Personalisierung: Ich verantwortete den Einstellungsbereich, inklusive der Implementierung der Dark-Mode-Funktion sowie der sicheren Anzeige von Nutzerdaten.

Anwenderdokumentation: Da ich als neues Teammitglied einen objektiven Blick auf die Software hatte, konnte ich eine besonders verständliche Anwenderdokumentation verfassen. Dies war ein entscheidender Faktor für die Qualitätssicherung, um die Usability-Anforderungen des Projekts zu erfüllen.

3. Fachlich korrekte Anwendung von Methoden und Techniken

Bei der Umsetzung meiner Aufgaben kamen fundierte Software-Engineering-Methoden zum Einsatz. Ich passte meine Arbeitsweise sofort an den bestehenden Scrum-Rhythmus an, schätzte meine Tasks realistisch ein und sorgte durch proaktive Kommunikation dafür, dass mein späterer Einstieg keine Blockaden für andere Teammitglieder verursachte.

Technisch arbeitete ich konsequent nach dem Prinzip des Component-Based Development. Beispielsweise entwickelte ich die Link-Elemente der Modulübersicht so abstrakt, dass sie an anderen Stellen der App leicht wiederverwendet werden konnten. In der Versionsverwaltung achtete ich auf sauberes Branching und aussagekräftige Commit-Messages gemäß den Git-Flow-Vorgaben des Teams, um Merge-Konflikte im bestehenden Repository zu vermeiden.

4. Verständnis der Zusammenhänge im Software Engineering

Der Erfolg meiner Arbeit basierte auf dem Verständnis, wie Frontend, Backend und Dokumentation ineinandergreifen. Besonders deutlich wurde dies bei der Datenvisualisierung: Hier musste ich die Modellierung im Backend nachvollziehen, um die Daten im Frontend performant darzustellen. Ich kommunizierte eng mit der Backend-Entwicklung, um die Struktur der API-Antworten abzustimmen.

Ein weiterer Fokus lag auf der Wartbarkeit. Da ich selbst fremden Code übernehmen musste, war mir bewusst, wie wichtig sauberer Code ist. Ich dokumentierte meine Funktionen direkt im Code und achtete auf eine klare Verzeichnisstruktur, damit zukünftige Entwickler meine Arbeit ebenso schnell verstehen können, wie ich die meiner Vorgänger.

5. Belegbarkeit der Beiträge

Meine Beiträge sind im Projekt durch die Repository-Historie (Commits zu Settings, Freundesliste und Overview) sowie durch das finale Nutzerhandbuch im Verzeichnis /deployment lückenlos dokumentiert und nachvollziehbar.