

M

C

Master Thesis

Skalierung einer Lern-Evaluationsplattform mit Serverless-Technologie

S

Master Thesis

Skalierung einer Lern-Evaluationsplattform mit Serverless-Technologie

von

Robert Nickel

zur Erlangung des Akademischen Grades

Master of Science

in Informatik, Schwerpunkt Software Engineering

an der Hochschule Konstanz für Technik, Wirtschaft und Gestaltung,

Matrikelnummer: 301337

Abgabedatum: 30. April 2022

Erstbetreuer: **Prof. Dr. Marko Boger**

Zweitbetreuer: **Prof. Jo Wickert**

Eine PDF Version dieser Thesis ist verfügbar auf <https://robertnickel.online/master-thesis/>.

Abstract

This thesis is about scaling a learning-evaluation-platform, using serverless technology. The term scaling refers to three different meanings. First, the learning-evaluation-platform should scale to many students without losing its educational usefulness. Second, scaling means to develop a product idea through methodical and economical guidelines to a product with an increasingly growing feature set. Third, the technology has to scale alongside the amount of users and requests, and always remain the state of responsiveness. Evaluating the set of serverless technologies, focussing on some of them in more detail, is the goal of this thesis. The focus of this investigation is the classification in the context of creating innovative digital products as a whole. Therefore, an examination of the currently available tools and frameworks is prefixed, their preconditions and potentials are evaluated, and a contextualisation of these principles compared to more established ones is made. The question of similarities and differences between serverless and reactive systems is answered. In order to make the results more practical, serverless technologies are used to build a learning-evaluation-platform called Lightbulb Learning¹, in order to show different aspects of these principles. Moreover, it is examined if and how the principles of serverless technology affect neighbouring topics such as the methodology of software development and business model evaluation. For a more complete assessment, that product should not only be the leading example of this thesis, but also be published and used by actual users. Thereby, even beyond technological aspects should be taken into consideration, so that the connection between serverless technology, development velocity, quality assurance, marketing and methodical aspect can be understood. It shows, that serverless technology can support the implementation of concepts such as lean startup, agile development and design thinking by massively reducing the amount of technological preconditions in order to reach the state of having a working system. Moreover it shows, that scaling a serverless system requires less complexity through the abstraction of generic challenges in that field.

¹<https://lightbulb-learning.io>

Inhaltsverzeichnis

1	Einleitung	1
2	Lernen durch Dialog	3
2.1	Ein Beispiel	3
2.2	Der Lehrauftrag	4
2.3	Der Evaluationsauftrag	4
2.4	Die Vision	5
2.5	Zielgruppen	6
2.5.1	Professoren, Lehrer und Kursautoren	6
2.5.2	Studenten, Schüler und Kursteilnehmer	6
2.6	Die Idee	7
2.6.1	Offene Fragen	7
2.6.2	Antworten auf offene Fragen	8
2.6.3	Feedback	9
2.6.4	Andere Typen von Lernartefakten	9
2.6.5	Crowdsourcing	10
2.6.6	Formatives und summatives Assessment	10
3	Methodik	13
3.1	Lean Startup	13
3.1.1	Die Bauen-Messen-Lernen Schleife	13
3.1.2	Minimal funktionsfähiges Produkt (MVP)	14
3.2	The Mom Test	15
3.3	Anforderungen an die Werkzeuge	16
3.3.1	Prinzip der Einfachheit	17
3.3.2	Lightbulb Learning als MVP	17
3.3.3	NoCSS Framework	17
4	Geschäftsmodelle	19
4.1	Geschäftsmodell A: Freemium	19
4.1.1	Mindelsee-Stipendium	20

4.2	Geschäftsmodell B: Affiliates-Programm	20
4.2.1	Geschäftsmodell von Udemy	21
4.2.2	Tests in Onlinekursen	21
4.2.3	Ergänzung um Lightbulb Learning	21
4.2.4	Mehrwert I: Steigerung der Lernqualität	22
4.2.5	Mehrwert II: Evaluationssystem und Zertifikate	22
4.2.6	Mehrwert III: Zusätzliche Einnahmequelle	23
4.2.7	Marketing und Vertrieb	23
5	Serverless-Technologie	25
5.1	Was ist Serverless-Technologie?	25
5.2	Frontend im Kontext von Serverless-Technologie	26
5.3	Function-as-a-Service (FaaS)	26
5.4	Reaktive Systeme	27
5.4.1	Antwortbereitschaft	27
5.4.2	Widerstandsfähigkeit	27
5.4.3	Elastizität	27
5.4.4	Nachrichtenorientiertheit	28
5.5	Reaktive Systeme und Serverless-Systeme	29
5.5.1	Gemeinsames Ziel: Antwortbereitschaft	29
5.5.2	Delegation	29
5.5.3	FaaS und Elastizität	30
6	Umsetzung: Lightbulb Learning	31
6.1	Interviews	31
6.1.1	Nutzerinterviews Studenten	31
6.1.2	Nutzerinterviews Professoren	33
6.1.3	Ergebnisse der Interviews	33
6.2	Technologie	34
6.3	Supabase	35
6.3.1	Vergleich: Supabase und Firebase	35
6.3.2	Relationale Datenbanken	36
6.3.3	PostgreSQL (alias Postgres)	36
6.3.4	Row-Level-Security	37
6.3.5	PostgreSQL Functions	37
6.3.6	PostgREST	39
6.4	Web Applikation mit SvelteKit	40
6.4.1	SvelteKit und Svelte	40
6.4.2	Multi-Page Apps (MPAs)	41

6.4.3	Single-Page Apps (SPAs)	41
6.4.4	Transitional Apps	42
6.4.5	Adapter, Server-Side Rendering und Hydration	43
6.5	TypeScript	43
6.6	Skalierung der Datenbank	44
6.7	Testing mit Playwright	45
6.8	Continuous Deployment Pipeline mit GitHub Actions	45
7	Evaluation	47
7.1	Experiment: Reaktive Systeme	47
7.2	Experimente mit AWS Lambda und Scala 3	49
7.2.1	Probleme mit AWS Amplify	49
7.2.2	Lambda Kaltstarts bei JVM-basierten Funktionen	49
7.2.3	FaaS API ist keine Aktoren API	51
7.2.4	Libraries und Dokumentation für Java	52
7.3	Svelte und SvelteKit	53
7.3.1	Intuitive Konzepte	53
7.3.2	Stolperstein: Serverseitige Authentifizierung	53
7.3.3	NodeJS Laufzeitumgebung ist kein Browser	54
7.3.4	PostgreSQL as a Service mit Supabase und lokale Entwicklung	54
7.3.5	Serverless nur für CRUD-Szenarien?	55
7.4	Serverless-Prinzipien	55
8	Fazit	57
8.1	Lightbulb Learning, ein reaktives System?	57
8.2	Zusammenhang zwischen Lean Startup und Serverless-Technologie	58
8.3	Zusammenhang zwischen agiler Entwicklung und Serverless-Technologie	58
8.4	Ist Serverless-Technologie die Zukunft der Apps?	59
8.5	Lightbulb Learning, eine Innovation?	59
8.5.1	Produktinnovation	60
8.5.2	Verfahrensinnovation	60
9	Ausblick	63
9.1	Technologischer Ausblick	63
9.1.1	Supabase Edge Functions mit Deno Deploy	63
9.2	Fachlicher Ausblick	64
9.2.1	Der Kontext der Nutzung	64
9.2.2	Das Hakenmodell	64
9.2.3	Legal-Themen	66

9.2.4	Lightbulb Learning Zertifikate	67
-------	--	----

Literatur		69
------------------	--	-----------

1

Einleitung

In dieser Thesis geht es um die Skalierung einer Lern-Evaluationsplattform mithilfe von von Serverless-Technologie¹. Der Begriff der Skalierung hat dabei drei wesentliche Bedeutungen. Erstens ist damit eine Skalierung der Lern-Evaluationsplattform als fachliche Lösung gemeint, welche auch für sehr große Kursgrößen funktionieren soll, ohne Einbüßung pädagogischer Sinnhaftigkeit. Die zweite Bedeutung ist die Skalierung von der Produktidee hin zu einem umfangreicher werdenden Produkt, mit einem Fokus auf wirtschaftliche und methodische Schwerpunkte. Drittens ist die Skalierung der Technologie gemeint, also die Sicherstellung, dass auch bei sehr vielen Zugriffen in kurzer Zeit das System vollständig antwortbereit bleibt. Die Betrachtung des Zusammenhang zwischen diesen drei Skalierungsdimensionen soll dabei helfen, ein vollständige Einordnung einiger Serverless-Technologien in den Kontext der Entwicklung von innovativen digitalen Produkten durchzuführen. Dafür geht eine Exploration einiger aktuell zur Verfügung stehenden Werkzeuge, deren Voraussetzungen und Potentiale sowie eine Kontextualisierung der Prinzipien im Vergleich zu üblichen Strukturen voran. Des Weiteren werden die Gemeinsamkeiten und Unterschiede zu reaktiven Systemen aufgezeigt. Um die jeweiligen Erkenntnisse greifbar zu machen, wird dazu unter Zuhilfenahme von Serverless-Technologie eine Lern-Evaluationsplattform namens Lightbulb Learning² entwickelt. Anhand dieser Plattform können unterschiedliche Aspekte dieser Prinzipien konkret aufgezeigt werden. Inwiefern diese technologischen Prinzipien auch die angrenzenden Bereiche wie die Methodik und die Geschäftsmodell-Evaluation beeinflussen, wird ebenfalls untersucht. Das dabei entstehende Produkt

¹Der Begriff Serverless lässt sich nicht präzise ins Deutsche übersetzen, da auch der Begriff Server üblicherweise nicht übersetzt wird.

²<https://lightbulb-learning.io>

soll nicht nur als Grundlage für diese Thesis dienen, sondern soll zusätzlich auch veröffentlicht werden. Dadurch soll eine genauere Einschätzung der Zusammenhänge der Produktentwicklung als Ganzes gewonnen werden können. Von der Serverless-Technologie selbst über die frühe Validierung der Funktionalität, die Entwicklungsgeschwindigkeit, die Aspekte der Qualitätssicherung und Vermarktung bis hin zu methodischen Prinzipien.

2

Lernen durch Dialog

Um einen Bezug zwischen der Serverless-Technologie und dessen Anwendung anhand eines Beispiels herzustellen, und die Evaluation der Technologie entsprechend einzuordnen, ist es sinnvoll, auch in den pädagogischen bzw. bildungswissenschaftlichen Aspekten einen Überblick zu haben. Darum geht es in diesem Kapitel.

2.1. Ein Beispiel

Für die Evaluation der Serverless-Prinzipien bedarf es eines Beispiels, welches möglichst viele typische Aspekte eines innovativen, digitalen Produkts abdeckt. Dazu gehören die Registrierung und der Login von Nutzern, öffentliche und private Informationen, dynamische und statische Daten und die damit verbundenen, regulierten Interaktionsmöglichkeiten. Der Zugang soll aus Nutzersicht durch einem beliebigen Browser und Gerät stattfinden. Eine im Rahmen des Moduls Entrepreneurship für Informatiker definierte Idee im Umfeld des E-Learnings bildet alle dieser Aspekte ab. Gleichzeitig ist die Idee, welche in den folgenden Abschnitten erläutert wird, ein interessanter Anwendungsfall, und soll einer ausführlichen Validierung in Form von Experimenten mit einem minimal funktionsfähigen Produkt (MVP)¹ unterzogen werden.

¹Siehe Abschnitt [3.1.2](#)

2.2. Der Lehrauftrag

Lehrer und Professoren haben die Aufgabe, Wissen zu vermitteln. Dafür gibt es bereits viele digitale Werkzeuge ganz unterschiedlicher Art, von denen bereits einige gut skalieren. Ein Beispiel ist Moodle², in dem Kurse angelegt und mit Inhalten, Referenzen, Videos, Abgabemöglichkeiten und Aufgaben gefüllt werden können. In den Moodle-Kursen sind in der Regel Menschen registriert, die sich außerhalb der Plattform im Rahmen des Kurses regelmäßig treffen, wie beispielsweise Schulklassen oder Hochschulkurse. Dazu passend sind die Nutzer auf der Plattform weder anonym noch pseudonym, sondern agiert mit seinem echten Namen. Ein weiteres Beispiel ist YouTube, welches ebenfalls für den Zweck der Wissensvermittlung genutzt werden kann. Die Harvard University hat einige Vorlesungsreihen vollständig auf diese Plattform hochgeladen, wie beispielsweise *Justice: What's the right thing to do?* von Michael Sandel, welche mit über 34 Millionen Aufrufen, mehr als 333.000 Likes und 16.000 Kommentaren³ auch außerhalb der Harvard University große Beachtung finden konnte. Durch die Verwendung dieser öffentlichen Plattform skaliert die Wissensvermittlung in Richtung der Konsumenten zwar besser, der Austausch über diese Inhalte und die kollaborative Erarbeitung ist hingegen kaum abbildbar. Lehrer stehen also vor der Wahl, öffentliche oder private Werkzeuge zu benutzen, wobei sich öffentliche Plattform durch die höhere Anonymität nur schwieriger strukturieren lassen, während weniger skalierbare Lösungen wie private Moodle Räume leichter zu strukturieren sind. Die Distribution der Struktur der Lerninhalte soll nicht bzw. nur indirekt in Lightbulb Learning abgebildet werden, daher lässt sich das Werkzeug mit anderen Werkzeugen, die genau dieses Ziel verfolgen, kombinieren.

2.3. Der Evaluationsauftrag

Neben der unidirektionalen Vermittlung von Wissen gehört es zum Lehrauftrag häufig auch dazu, bei den Schülern oder Studenten zu beurteilen, ob das Gelernte tatsächlich verstanden wurde und angewendet werden kann. Dies wird im Folgenden als Lernevaluation bezeichnet. Hierfür nutzen Lehrer und Professoren Prüfungen, diese sind typischerweise Klausuren, oder auch alternative Leistungsnachweise in Form von Projekten, Präsentationen oder Ausarbeitungen. Diese können je nach Kontext durchaus pädagogisch wertvoller sein als reguläre, schriftliche Klausuren, gehen aber gleich-

²<https://moodle.org/>

³Siehe <https://youtu.be/kBdfcR-8hEY>

zeitig mit dem Problem der fehlenden Skalierbarkeit einher. Aus diesem Grund der Praktikabilität kommt für größere Kurse häufig nur eine Klausur in Frage, welche die zunehmende Skalierbarkeit mit pädagogischer Sinnhaftigkeit bezahlt. Besonders stark ist dieser Effekt bei Multiple-Choice-Klausuren zu beobachten. Diese können neben der Variante auf Papier auch mit digitalen Werkzeugen abgebildet und automatisiert ausgewertet werden. Jedoch trainiert die Vorbereitung auf Multiple-Choice-Klausuren häufig die Fähigkeit, wiederzugeben wie die richtigen Antworten aussehen, und nicht, was deren inhaltliche Bedeutung ist [Pro22]. Aus Sicht eines Prüflings gestaltet sich die Optimierung auf eine solche Prüfung also nicht durch das langfristige Erlernen und Verstehen von Zusammenhängen, sondern durch das kurzfristige Auswendiglernen von Formulierungen. Hinzu kommen wahrscheinlichkeitstheoretische Effekte, insbesondere unter Einbeziehung des Ausschlussverfahrens, welche zu einer sinkenden Validität eines solchen Tests führen. Dies "verringert die Akzeptanz des Tests bzw. reduziert dessen Image im Bezug auf wiss. Seriosität und Verbindlichkeit", fasst [Pro22, unter (4)] zusammen. Die Multiple-Choice-Prüfung ist ein besonders extremes Beispiel steigender Skalierbarkeit, welche mit sinkendem pädagogischem Sinn abgegolten wird.

2.4. Die Vision

Die Vision von Lightbulb Learning ist die Erschaffung eines pädagogisch vertretbaren Lern-Evaluationssystems, welches gleichzeitig skaliert. Die Kontexte, in denen ein solches Werkzeug eingesetzt werden könnte, sind die Schulbildung, die Hochschulbildung und Onlinekurse. Da unterschiedliche Rollen am Prozess partizipieren, sind für ein gesamtheitliches Verständnis auch die wichtigsten Perspektiven zu betrachten: Die des Professors (oder Lehrers oder Kursautors), und die des Studenten (oder Schülers oder Kursteilnehmers), welche im Folgenden jeweils austauschbar verwendet werden. Da das Lern-Evaluationssystem in unterschiedlichen Kontexten unterschiedliche Zielgruppen bedient, wird eine Festlegung auf die Begriffe eines einzigen Kontexts nicht den Anforderungen an das Systems gerecht.

2.5. Zielgruppen

2.5.1. Professoren, Lehrer und Kursautoren

Aus Sicht der Professoren soll dieses System dabei unterstützen, qualitative Evaluationen der studentischen Leistungen zu erstellen, während der Aufwand für diese Tätigkeit pro Student möglichst gering gehalten werden soll, um möglichst viele Studenten bewerten zu können und damit die Skalierbarkeit aufrecht zu erhalten. Im akademischen Kontext ist das Wertversprechen einer Lösung, welche diese Anforderungen erfüllt, ein Zeitgewinn für den Professor. Wendet man diese Systemanforderungen außerhalb des akademischen Kontextes an, so könnten beispielsweise Autoren von Online-Kursen dieses Werkzeug verwenden, um denjenigen Kursteilnehmern, die einen bestimmten inhaltlichen Fortschritt nachweisen können, ein entsprechendes Zertifikat ausstellen. Somit lässt sich festhalten, dass die Anforderungen an das System aus Sicht dieser Rolle vor allem Skalierbarkeit und Evaluationsqualität sind.

2.5.2. Studenten, Schüler und Kursteilnehmer

In diesem Abschnitt soll die Perspektive des Lernalers eingenommen werden, um dessen Prioritäten, Bedürfnisse und Wünsche zu beleuchten. So ist beispielsweise im klassischen akademischen Kontext die studentische Perspektive auf die Frage der Bewertung eine völlig andere als die des Professors. Es ist besonders wichtig, fair und valide beurteilt zu werden, und in einem System zu lernen, indem langfristiges Lernen⁴ belohnt wird. Die Fairness ist ein Aspekt, der vor allem aus dem sozialen Umfeld der Studenten entsteht und nicht in jedem Kontext die gleiche Bedeutung hat, da etwa für Onlinekurse möglicherweise kein Bezug zu anderen Teilnehmern besteht, welche einen direkten Vergleich überhaupt ermöglichen würden. Für diese Fälle ist der Begriff der Fairness eher im Sinne der Nachvollziehbarkeit zu deuten. Die Validität rückt aus der Sicht der Lernalers häufig erst nach der Prüfung in den Vordergrund: Hat jemand eine Prüfung bestanden und eine entsprechende Note dafür erhalten, so ist es aus Sicht des Absolventen von zentraler Bedeutung, dass das Bestehen eine mit Aufwand und Fähigkeiten erarbeitete, nicht inflationäre Leistung ist, und auch von der Außenwelt als solche wahrgenommen wird. Im Kontext eines Onlinekurses spielt die Bedeutung der Validität eine besondere Rolle, da sie besonders gefährdet ist. Liegen beispielsweise die Antworten auf einen Fragekatalog im Internet bereits öffentlich vor, so kann die Validität der Leistung der Beantwortung dieser Fragen selbst dann in Mitleidenschaft gezogen werden,

⁴als Gegensatz zum kurzfristigen Auswendiglernen

wenn die fertigen Antworten nicht verwendet wurden, da dies durch das unkontrollierbare Umfeld des Prüflings nicht ausgeschlossen werden kann. Zusammenfassend lässt sich sagen, dass aus der Perspektive der Lerner insbesondere die Fairness, die Validität und der Rahmen, der langfristiges Lernen fördert, von zentraler Bedeutung sind.

2.6. Die Idee

Die beschriebenen Anforderungen sollen mittels eines innovativen digitalen Werkzeugs adressiert und umgesetzt werden. Der Kern für die Umsetzung der beschriebenen Anforderungen ist das Crowdsourcing bzw. die Schwarmauslagerung. Studenten erhalten den Auftrag, selbst Fragen zu formulieren, die Fragen der anderen Studenten zu diskutieren und zu beantworten, und sich gegenseitig Feedback zu geben. Dabei ist jede Facette dieser Leistung, also jede Form der Interaktion, ein Teil der Gesamtleistung, was eine pädagogisch vertretbare Form der gegenseitigen Wissensvermittlung und individuellen Wissensaufnahme ermöglicht. Zeitgleich ermöglicht sie detaillierte Einblicke in die Entwicklung des Wissensstands eines Studenten über einen längeren Zeitraum, beispielsweise über das gesamte Semester. Ein Professor kann die Übersicht sämtlicher Leistungen eines Studenten in chronologischer Reihenfolge zu jeder Zeit einsehen, was einen guten Eindruck über das Leistungsniveau des Studenten vermittelt. So ist nicht nur frühes Feedback in Richtung des Studenten und die Möglichkeit der Verbesserung fehlender Leistung möglich, sondern wird auch dem Professor widergespiegelt, auf welchem Niveau der Kurs als Ganzes sich befindet. Dies kann (und sollte) Auswirkungen auf die Lehre haben, um dem in Abschnitt 2.2 bereits beschriebenen Lehrauftrag möglichst gut nachzukommen. Auch für den Kontext der Onlinekurse greifen die gleichen Mechanismen. Durch die schnelle Bewertbarkeit kann ein Kursautor einen interaktiveren Umgang mit seinen Kursteilnehmern haben und diese individuell bewerten, was vorher nicht möglich gewesen wäre. Weshalb die Interaktion der Prüflinge untereinander einen guten Überblick über das Leistungsniveau der partizipierenden Individuen gibt, wird in den folgenden Kapiteln beschrieben.

2.6.1. Offene Fragen

Insbesondere die Fähigkeit gute, originelle und offene Fragen zu einem bestimmten Themenbereich zu stellen, bezeugt sehr stark die Tiefe des Verständnisses, welche der Verfasser der Frage zu dem Zeitpunkt hat. Diese Aussage stützt sich auf die Begründung, dass beim Erstellen einer Frage immer der Kontext, der Problemraum und

die Relevanz der Frage definiert werden muss, während eine Antwort typischerweise diesen Kontext lediglich aufgreift. Ein einfaches Beispiel aus dem Themenbereich des Quantencomputings könnte lauten: „Was sind die wesentlichen Unterschiede zwischen einem Qubit und einem normalen Bit?“. Um eine solche Frage zu stellen, muss dem Fragesteller klar sein, dass es einen Unterschied gibt, und die Frage nach dem Unterschied aus seiner Sicht eine bedeutende Rolle für das Verständnis von Qubits, oder im weiteren Sinne von Quantencomputing, spielt. Der Antwortgeber hingegen bekommt den Problemraum vorgelegt, und muss innerhalb dessen lediglich eine Antwort formulieren. Gleichzeitig ermöglichen gute Fragen eine qualitativ hochwertige Diskussion durch die Beantwortung - der wiederum einen Mehrwert für alle Beteiligten darstellt. Somit kann nicht nur Verständnis und dessen Kontext durch eine gute Frage ausgedrückt werden, sondern werden auch die Fähigkeiten der Kommunikation, insbesondere der Moderation, trainiert, welche in der modernen Arbeitswelt von zentraler Bedeutung sind.

2.6.2. Antworten auf offene Fragen

Die Funktion des Arbeitsauftrags einer Antwort ist naheliegend: es wird der Umgang mit den Werkzeugen der Problemlösung, beispielsweise der Internetrecherche oder Lernvideos, sowie der Adaption von theoretischem Wissen auf einen konkreten Kontext trainiert und bewertet. Neben dem Leistungsnachweis, Fragen korrekt und präzise beantworten zu können, greifen noch weitere Mechanismen, wie die Externalisierung im Sinne des SECI-Modells⁵ [Non97]. Damit ist gemeint, dass implizites Wissen zu explizitem Wissen transformiert wird, indem es formuliert wird. Im darauffolgenden Schritt der Kombination im Rahmen der Diskussion der Frage bzw. einer bestimmten Antwort kann dieses von anderen Studenten durch die Internalisierung wiederum zu implizitem Wissen und daher angewendet werden. Gleichzeitig wirkt die Antwort als Feedback an den Fragesteller. Beispielsweise sind Ein-Wort-Antworten, welche eine Frage korrekt und präzise beantworten, gute Hinweise darauf, dass die Qualität bzw. die Offenheit dieser Frage eher gering war. Ist eine Frage zu unklar formuliert, so könnten Antworten von unterschiedlichen Studenten diese stark unterschiedlich beantworten. Erhält eine Frage hingegen mehrere, ausführliche Antworten, welche im Laufe der Zeit mehrfach korrigiert und präzisiert werden, so ist die Wahrscheinlichkeit, dass es sich um eine gute Frage handelt, deutlich erhöht. Wurde eine Antwort einmal gegeben, so muss diese nicht endgültig sein.

⁵SECI = Sozialisation, Externalisierung, Kombination, Internalisierung

2.6.3. Feedback

Erhält der Verfasser einer Antwort Feedback darauf, so kann er seine Antwort auf die Frage im Sinne der Korrektur und Präzisierung verbessern. Gleichzeitig ist auch der Prozess des gegenseitigen Feedback-Gebens für den Verfasser des Feedbacks hilfreich. Der Empfänger des Feedbacks erhält die Gelegenheit, Aspekte in seine Antwort einzubeziehen, welche er vorher nicht in Erwägung gezogen hatte, oder auch klare Fehler zu korrigieren. Da er gleichzeitig aber auch im vollen Bewusstsein dessen ist, dass der Feedback-Geber selbst nicht zwangsläufig ein Experte für den behandelten Themenbereich sein muss, muss er ein eigenes Urteil für die Einordnung und den Umgang mit dem Feedback bilden. Durch die Augenhöhe des Feedbacks steht der Verbesserungsprozess eher im Vordergrund als die Bewertung der Leistung, denn diese wird von einer anderen Person als vom Feedbackgeber durchgeführt. Daraus geht hervor, dass sich der Fokus mehr auf die Verbesserung der Inhalte selbst konzentriert, und weniger auf die Einordnung des Feedbacks als Bewertung. Der Feedbackgeber trainiert seinerseits ebenfalls eine Kompetenz, welche in einem Arbeitsleben, in dem in kleinen Teams eng zusammengearbeitet wird, zunehmend wichtiger wird. So kann mit Feedback etwa nicht nur Korrekturbedarf, sondern auch Wertschätzung ausgedrückt werden. Auch Coursera, eine E-Learning-Plattform, welche in Abschnitt 4.2.2 noch genauer beschrieben wird, verwendet gegenseitiges Feedback, geht hier allerdings noch weiter und fordert die Kursteilnehmer sogar auf, sich gegenseitig eine Note zu geben [Cou22b].

2.6.4. Andere Typen von Lernartefakten

Die beschriebenen offenen Fragen, deren Antworten sowie Feedback auf diese Antworten bilden nur einen kleinen Teil der Formen ab, die ein Lernartefakt annehmen kann. Einige weitere Beispiele wären Programmiercode, Mathematikaufgaben bzw. deren Lösungen, Visualisierungen wie Graphen, Skizzen oder Pläne, Aufsätze, Bilder, Videos oder Tonaufnahmen. Technisch gesehen wäre es möglich, diese ebenfalls in die Entwicklung von Lightbulb Learning einzubeziehen. Dennoch wird für den Kontext dieser Thesis der Fokus auf den Aufgabentyp *offene Frage* gelegt, da dieser einen Kompromiss zwischen der Einfachheit der Umsetzung, was eine schnelle Validierung durch Nutzer ermöglicht, und der Breite der Anwendungsmöglichkeiten darstellt.

2.6.5. Crowdsourcing

Lightbulb Learning beinhaltet selbst keine Intelligenz in der Hinsicht der Einordnung, Strukturierung oder Filterung von Inhalten, sondern stellt lediglich ein Werkzeug für Kurse bereit, dies kollaborativ erledigen können. Es stellt sich allerdings die Frage, ob das System durch nicht-technische Angriffe wie Absprachen manipulierbar ist. Ein mögliches Szenario für einen solchen Versuch wäre die Clusterbildung. Damit ist gemeint, dass Leute sich eher an den Teilen der Diskussion beteiligen, die von ihren jeweiligen Bekannten dominiert werden. Über Kommunikationskanäle außerhalb von Lightbulb Learning könnte sinngemäß etwa vereinbart werden, dass sich jeder ein paar Fragen überlegt, den anderen die Antworten schickt und diese die Antworten, ohne dabei das entsprechende Verständnis aufzubauen, nur noch in das System eintragen. Das ist prinzipiell möglich, aber für die Erlangung einer guten Bewertung nicht sinnvoll und leicht zu entdecken. Inhaltlich gleiche Antworten oder gleiche Formulierung auf eine Frage können als Hinweise aufgefasst werden. Zusätzlich wäre es auch möglich, eine Analyse des Interaktionsgraphen durchzuführen, welche die Clusterbildung erkennt und den Professor auf Anomalien hinweist. Insbesondere in Hinsicht der Nutzung des Like-Features für Fragen und Antworten hätte eine solche Analyse eine besondere Aussagekraft. Dabei ist zu jedoch auch zu beachten, dass Clusterbildung nicht unbedingt ein Missbrauchsversuch ist, sondern auch schlicht Überlappungen von Interessensbereichen beschreiben kann. Insbesondere können solche Cluster auch als Lerngruppen begriffen werden, welche durch den Austausch über die Inhalte eines Kurses aktiv dazu beitragen, die eigene Lernqualität zu erhöhen.

2.6.6. Formatives und summatives Assessment

Bei Prüfungen unterschiedlicher Art, wie sie heutzutage im akademischen Kontext gängig sind, handelt es sich in den meisten Fällen um summatives Assessment. Damit ist gemeint, dass beispielsweise durch eine schriftliche oder mündliche Klausur am Ende eines Semesters eine Wissensabfrage mit dem Ziel der Lernevaluation für den gesamten Zeitraum erfolgt. Formatives Assessment gibt im Gegensatz dazu über die gesamte Laufzeit des Semesters Einblicke in den aktuellen Lernstand der Studenten, so dass die Lernkurve durch individuelle Intervention korrigiert werden kann. Dieser Ansatz ist somit für den Professor deutlich aufwendiger und kann nur in kleineren Kursgrößen angewendet werden, erhöht aber sowohl die Lern- als auch die Evaluationsqualität. Eine passende Metapher für summatives Assessment ist das Foto vom Zieleinlauf des NYC Marathons in Abbildung 2.1. Hier wird die gesamte Leistung des Absolvierens eines Marathons auf eine einzige, sehr eindimensionale Metrik vereinfacht. Auch wenn

dieser Ansatz seine Daseinsberechtigung hat, nämlich in der Frage, wer am schnellsten 42,195 Kilometer zu Fuß zurücklegen kann, fallen Aspekte wie die Körpergröße, das Gewicht, Verletzungen auf den letzten Kilometern oder die Qualität der Ausrüstung überhaupt nicht ins Gewicht.



Abbildung 2.1: Zieleinlauf des NYC Marathons 2018

Quelle: <https://www.nyrr.org/tcsnycmarathon/getinspired/photos-and-stories/marathon-goals-virtual-training>

Lightbulb Learning verfolgt das Ziel, eine Plattform für die Skalierung von formativem Assessment zu sein. In der Marathon-Metapher entspräche dies einem vollständigen Streckenprotokoll der Läufer mit mehreren Checkpoints, wie in Abbildung 2.2 dargestellt. Im übertragenen Sinne sind dabei mehr unterschiedliche Messpunkte und die Messung über einen längeren Zeitraum entscheidend.

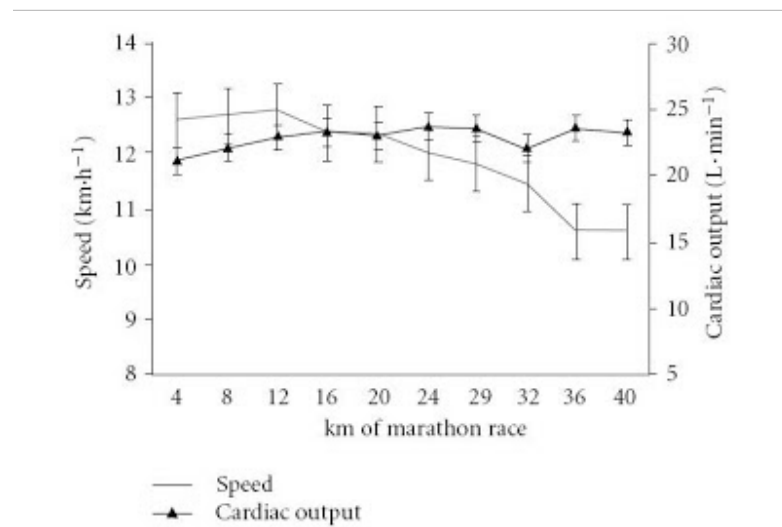


Abbildung 2.2: Streckenprotokoll über Londoner Marathon

Quelle: <https://ultrastu.blogspot.com/2013/05/the-negative-split-fallacy-part-2.html>

3

Methodik

Neben der Evaluation von Serverless-Technologie erfüllt Lightbulb Learning auch die Funktion, Erfahrungen mit den methodischen Prinzipien moderner Unternehmensgründung durch deren Anwendung zu sammeln und diese zu bewerten. Dafür werden die Prinzipien einiger Modelle aus diesem Kontext kurz zusammengefasst und deren Umsetzung im Kontext von Lightbulb Learning beschrieben.

3.1. Lean Startup

3.1.1. Die Bauen-Messen-Lernen Schleife

The Lean Startup ist ein Buch von Eric Ries aus dem Jahre 2011 [vgl. [Rie11](#)], welches die Prinzipien der Lean Manufacturing aus der Toyota Kata [vgl. [Rot09](#)], in Kombination mit einigen Prinzipien wie der agilen Entwicklung und dem Design Thinking kombiniert und in den Kontext der schlanken Unternehmensgründung überführt. Im Vordergrund steht der Ansatz, aus Unternehmenssicht validiertes Lernen in den Mittelpunkt zu stellen. Dabei liegt ein besonderes Augenmerk auf dem Weglassen von allem, was nicht für die Validierung benötigt wird, um schnelleres, realistischeres Feedback anstelle von Annahmen als Grundlage für Entscheidungen zurate ziehen zu können. Durch kontinuierliche Weiterentwicklung der Ideen, Experimenten mit realen Kunden und den daraus ableitbaren Schlussfolgerungen soll wiederum das Produkt korrigiert und weiterentwickelt werden. Im Mittelpunkt von Lean Startup steht der Ansatz, welcher als Bauen-Messen-Lernen Zyklus [[Rie11](#), S. 81] bezeichnet wird.

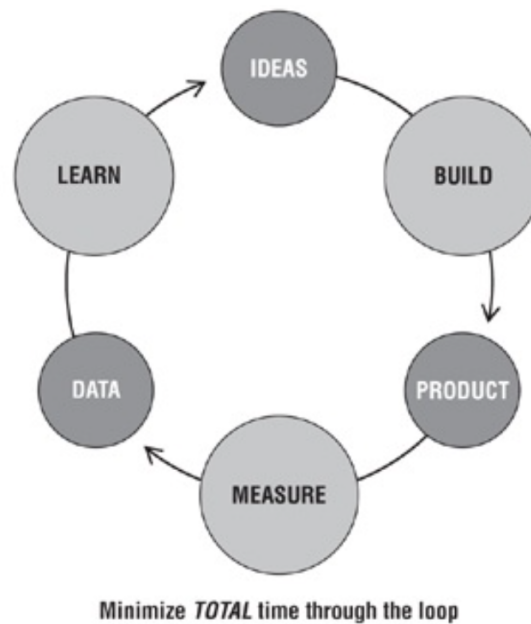


Abbildung 3.1: Build - Measure - Learn Loop, [Rie11, S. 81].

In Abbildung 3.1 ist dieser Zyklus visuell dargestellt. Die größeren, helleren Kreise beschreiben dabei Aktivitäten, während die kleineren, dunkleren Kreise Artefakte symbolisieren. Der Beginn des Zyklus ist eine vermutete Hypothese oder Idee zu finden, welche als Ausgangspunkt für die weitere Validierung dient. Diese Idee wird im Bauenschnitt (build) umgesetzt, dabei ist der Fokus auf der Identifikation des größten Risikos, der Planung eines Tests und der Umsetzung einer Variante der Lösung, die lediglich gut genug ist, um die Idee zu validieren. Im Messen-Schritt wird so objektiv wie möglich untersucht, ob die initiale Hypothese standhalten konnte, und überführt Beobachtungen, Messungen und Ergebnisse in auswertbare Daten. Diese werden dann im letzten Schritt genau analysiert, um eine Einschätzung der Hypothese vorzunehmen, und diese als Grundlage für eine weitere Iteration mit angepasster Hypothese verwendet. Durch dieses Vorgehen können mutige und innovative Technologie- oder Geschäftsentscheidungen getroffen und dabei gleichzeitig das Risiko für Fehlentscheidungen reduziert werden.

3.1.2. Minimal funktionsfähiges Produkt (MVP)

Ein Medium für die Umsetzung des Ansatzes ist das minimal funktionsfähige Produkt [vgl. Rie11, S. 96], wobei typischerweise die englische Bezeichnung Minimum Viable Product bzw. dessen Akronym MVP verwendet wird. Die Bedeutung des MVP ist, dass eine Produktidee so lange auf den wesentlichen Kern reduziert wird, bis der Umfang

so klein ist, das mit wenig Aufwand und somit geringem Risiko validiert werden kann, ob eine Idee funktioniert oder nicht. Scheitert eine Idee so besteht der Erfolg darin, dies früh erkannt zu haben, und somit weitere Ressourcen wie Zeit und Geld nicht wegen dem fehlenden Wissen um den Umstand des Scheiterns vergeudet zu haben. Funktioniert hingegen eine Idee, so kann auch dies früh festgestellt und entsprechend investiert und weiterentwickelt werden. Lightbulb Learning soll wie in dem Buch von Eric Ries beschrieben als MVP entwickelt und früh veröffentlicht werden. Somit soll früh gelernt werden, ob die Idee funktioniert. Tut sie das nicht, so können Änderungen und Experimente durchgeführt werden, welche ihrerseits evaluiert werden und zu weiteren Schlussfolgerungen führen können.

3.2. The Mom Test

Die von Robert Fitzpatrick in The Mom Test [vgl. [Fit13](#)] beschriebene Methode beschäftigt sich mit der Frage, wie Nutzerinterviews vor der Entwicklung einer Geschäftsidee so strukturiert werden können, dass deren Resultate eine möglichst hohe Objektivität aufweisen. Dabei wird ein besonderer Fokus darauf gelegt, dass die Aussagen von Gesprächspartnern häufig dahingehend verzerrt werden, dass der Interviewer in seinen Annahmen möglichst bestätigt wird. Die Begründung dafür ist, dass vor allem nahestehende Leute, teilweise unterbewusst, lieber etwas Nettes sagen als die Wahrheit. Es werden Techniken beschrieben, die diesen Interpretationsspielraum verringern, um ein genaueres Bild vor der Entscheidung für eine größere Investition von Zeit oder Geld zu erhalten. Die drei hervorgehobenen Prinzipien des Buches lauten:

1. „Rede über ihr Leben, und nicht über deine Idee.
2. Frage nach konkreten Dingen der Vergangenheit, und nicht nach generischen Dingen oder Meinungen über die Zukunft.
3. Rede weniger, und höre mehr zu.“ [[Fit13](#), S. 128]

Um diese Prinzipien anzuwenden, wird beispielsweise vorgeschlagen, in einem Nutzerinterview vorerst gar nicht zu erwähnen, was die eigene Idee ist. Selbst die Information, dass es sich gerade um ein Nutzerinterview handelt, könnte erst im späteren Verlauf andeuten werden, um detaillierte Einblicke in das Leben des Gesprächspartners zu erhalten. Um das zweite Prinzip anzuwenden, könnte beispielsweise nach aktuellen Lösungsstrategien für Probleme gefragt werden, da die subjektive Einschätzung der Größe eines Problems eine Aussage über die Bereitschaft der Nutzung beinhaltet.

Sagt ein Gesprächspartner beispielsweise, dass es für ihn ein riesiges Problem ist, und er bereits nach Lösungen dafür gesucht hat, aber noch nicht fündig geworden ist, so kann dies als Bestätigung für die Wahl des richtigen Problems gedeutet werden. Um sicher zu gehen, dass die Problematik wirklich groß genug ist, sollte nach dem zweiten Prinzip nach einer konkreten Investition gefragt werden. Diese muss nicht finanzieller Natur sein, sondern kann auch aus Zeit oder sozialem Aufwand bestehen. Ist ein Gesprächspartner nicht bereit, diese Hürden auf sich zu nehmen, so ist dies wiederum ein starker Hinweis dafür, dass das Problem tatsächlich nicht besonders groß für ihn ist. Mit dem dritten Prinzip wird der Hang adressiert, in eine Diskussion zu verfallen, sich zu rechtfertigen oder sehr stark in den Verkaufsmodus überzugehen. Hier wird unter anderem dazu geraten, am Ende eines Nutzerinterviews, in dem es um die Lebenswelt des Gesprächspartners geht, eine klare Trennung zu machen, und fragend zu formulieren, ob er sich für die eigene Idee interessiert. Dadurch können Anknüpfungspunkte für den späteren Vertrieb des Produkts entstehen, ohne dabei das Verlassen der Objektivität im vorhergehenden Gesprächsabschnitt zu riskieren.

3.3. Anforderungen an die Werkzeuge

In diesem Abschnitt geht es um die Anforderungen, welche die ausgewählten Werkzeuge wie Technologien und Methoden erfüllen müssen, um für die Entwicklung von Lightbulb Learning infrage zu kommen. Die Rahmenbedingungen dafür sind die Entwicklung eines vollständigen digitalen Produkts, von der Entwicklung der Idee, über die Ausarbeitung des Produktdesigns bis hin zur Umsetzung, Qualitätssicherung und dem Betrieb der Software. In den Bereich der Qualitätssicherung fällt sowohl das Schreiben von automatisierten Tests, welche einen definierten Funktionsumfang abdecken, aber auch die Validierung der Idee selbst, der Usability, der Datensicherheit und des Geschäftsmodells. Entsprechend dem breiten Spektrum der Problemfelder wird ein breites Spektrum an Werkzeugen genutzt. Auch der Umstand, dass sich der persönliche Rahmen auf eine Person beschränkt, und somit die Verteilung der Kompetenzen auf die Breite und folglich nicht auf die Tiefe bedingt, konkretisiert die Anforderungen an die verwendeten Werkzeuge. Eine weitere Dimension für die Anforderungsanalyse ist die zeitliche: Innerhalb einiger Monate wird ein lauffähiges, validiertes und sinnvolles minimal-funktionsfähiges Produkt entwickelt, was eigene Limitationen mit sich bringt. Es lässt sich zusammenfassen: Alle verwendeten Werkzeuge müssen mit einer steilen Lernkurve einher gehen, eine hohe Kompatibilität zueinander aufweisen und in ihrer Summe das gesamte Spektrum von technologischen, methodischen, gestalterischen

oder wirtschaftlichen Problemfeldern abdecken.

3.3.1. Prinzip der Einfachheit

Die verwendeten Werkzeuge fußen allesamt auf den agilen Prinzipien [vgl. [Bec01](#)], allen voran dem zehnten agilen Prinzip: „Einfachheit – die Kunst, die Menge nicht getaner Arbeit zu maximieren – ist essenziell“ [[Bec01](#), 10. Prinzip]. Unter der Zuhilfenahme möglichst guter Werkzeuge für die Aufwandsminimierung sollen die Anforderungen zufriedenstellend erfüllt werden.

3.3.2. Lightbulb Learning als MVP

Der Fokus auf den Aufgabentypen *offene Frage* ist im Lichte des Prinzips der Einfachheit zu sehen: Es ermöglicht eine Validierung des Produkts, erfordert aber nicht die Umsetzung größerer Komplexität wie es beispielsweise automatisiert bewertbare Programmieraufgaben oder Dateiuploads tun würden. Ab einem möglichst frühen Zeitpunkt soll eine funktionierende Version des Produkts online zur Verfügung zu stehen, welches von Nutzern ausprobiert werden kann. Dieser Fokus ist ein Resultat aus dem ersten der zwölf agilen Prinzipien: „Unsere höchste Priorität ist es, den Kunden durch frühe und kontinuierliche Auslieferung wertvoller Software zufrieden zu stellen“ [[Bec01](#), 1. Prinzip]. Der Fokus auf funktionierende Software als das wichtigste Fortschrittsmaß führt zu einer realistischen Einschätzung des aktuellen Stands und der verbleibenden Ziele im Sinne des Fortschritts der Thesis und des Produkts Lightbulb Learning, und der Korrektheit der Richtung. Technologisch wird Serverless-Technologie eingesetzt, da diese durch technische Abstraktion die Schwelle hin zu funktionierender und bereitgestellter Software senkt. So sollen mit Serverless-Technologien beispielsweise die Themen der Benutzerverwaltung, der Datenpersistenz, der Sicherheit und Zugreifbarkeit auf diese Daten sowie der Aufbau einer Continuous Deployment Pipeline, das Hosting einer Web-Applikation und die Skalierung all dieser Aspekte abgebildet werden. Somit kann die Idee validiert werden, ohne langwierige Entwicklung oder hohe Entwicklungskosten durch größere Teams in Kauf nehmen zu müssen. Eine detaillierte Beschreibung evaluierter und verwendeter Serverless-Komponenten befindet sich in Kapitel 5.

3.3.3. NoCSS Framework

Eine weitere Anforderung an die Entwicklung von Lightbulb Learning ist die Entwicklung einer Designsprache, woraus die Anforderung an das Werkzeug resultiert, diese

möglichst einfach, konsistenz und allgemeingültig zu erstellen und weiterzuentwickeln. Die Form dieser Designsprache soll dabei explizit über Vektorgrafiken in Designwerkzeugen wie Figma oder Sketch hinausgehen, und stattdessen einsatzfähiger HTML und CSS Code sein. Die Klasse an Frameworks, welche diese Anforderungen erfüllt, werden NoCSS-Frameworks genannt.

4

Geschäftsmodelle

In diesem Kapitel geht es um zwei unterschiedliche Geschäftsmodelle, welche während der Entwicklung des Produkts entstanden sind. Das erste Modell fokussiert dabei insbesondere den akademischen Sektor, während das zweite sich auf Onlinekurse konzentriert.

4.1. Geschäftsmodell A: Freemium

Von Beginn des Projekts an war Lightbulb Learning als Lern- und Prüfungsplattform vor allem für den akademischen Sektor, also für Universitäten und Fachhochschulen gedacht. Da die Zielgruppe das Geschäftsmodell wesentlich beeinflusst, besteht es aus der Monetarisierung durch die Professoren dieser Hochschulen bzw. im zweiten Schritt durch die Hochschulen selbst. Mit einem Freemium-Modell könnten so bestimmte Features oder eine begrenzte Kursgröße kostenfrei für den Einstieg zur Verfügung gestellt werden. Dieser könnte dann gegen einen festen monatlichen Preis um weitere Features und unbegrenzte Kursgrößen ergänzt werden, welcher pro Professor anfällt. Da das Wertversprechen eine echte Zeitersparnis der Professoren beinhaltet, galt die Annahme einer gewissen Zahlungsbereitschaft. Diese könnte entweder aus Budgets für digitale Werkzeuge der Professoren bestritten werden, im Zweifelsfall aber auch privat bezahlt werden. Würde ein bestimmter Schwellwert an Professoren der gleichen Hochschule eine Lightbulb Learning Lizenz beschaffen, so könnte eine insgesamt günstigere, hochschulweite Lizenz angeschafft werden, welche alle Premium Funktionen von Lightbulb Learning für alle Professoren und alle Studenten dieser Hochschule freischaltet.

In dem Fall wäre auch eine Integration der Authentifizierungsschnittstelle denkbar, so dass die Zuordnung der Nutzer zu den echten Personen noch direkter ist. Durch die hochschulweite Premiumversion sollten auch diejenigen Professoren das Werkzeug kennenlernen, welche von sich aus keine Zahlungsbereitschaft gezeigt hätten, mit dem Ziel, weitere Professoren und Hochschulen von der Idee zu überzeugen, und somit einen Netzwerkeffekt in Gang zu setzen.

4.1.1. Mindelsee-Stipendium

Für die Finanzierung weiterer Zeit, in welcher weiter an Lightbulb Learning gearbeitet werden kann, wurde ein Businessplan für die Bewerbung beim Mindelsee-Stipendium erarbeitet. Dieser Businessplan enthält den zum Zeitpunkt der Bewerbung aktuellen Stand der Idee von Lightbulb Learning, eine Zielgruppen- und Marktanalyse, das Geschäftsmodell, die Finanzplanung und eine Chancen- und Risikoabwägung. Aus der Absage der Bewerbung konnte werthaltiges Feedback bezogen werden, woraus, insbesondere im Bezug auf das Geschäftsmodell, eine zweite Option entwickelt werden konnte. Da die Beschaffung von Software an Hochschulen teilweise bis auf Landesebene institutionell organisiert ist, sorgen die langen Entscheidungswege und die ausgiebige technologische Evaluation für eine gewisse Trägheit in der Adaption innovativer Software. Dies führt zur Erschwerung der Eindringung in diesen Markt, so dass die Risiken dieses Ansatzes zu hoch für eine ernsthafte Weiterverfolgung sind.

4.2. Geschäftsmodell B: Affiliates-Programm

In diesem Geschäftsmodell sind die primäre Zielgruppe, anstelle von Professoren und deren Studenten, Autoren von Onlinekursen und deren Teilnehmer. Ein Blick auf die aktuellen Zusammenhänge zwischen diesen Kursautoren und den Plattformen, auf denen sie ihre Kurse anbieten, eröffnet einen guten Zugang zum Mehrwert durch Lightbulb Learning. Einige Beispiele für Plattformen, auf denen Kursautoren ihre Kurse verkaufen, sind Udemy¹, Coursera², LinkedIn Learning³ oder Academind⁴.

¹<https://www.udemy.com/de/>

²<https://www.coursera.org/>

³<https://www.linkedin.com/learning>

⁴<https://academind.com/>

4.2.1. Geschäftsmodell von Ude-my

In diesem Abschnitt soll eine genauere Untersuchung des Geschäftsmodells von Ude-my erfolgen. Hier werden Kurse von Kunden gekauft und können daraufhin, in der Regel in erster Linie in Form von Videos, von diesen zu jedem beliebigen Zeitpunkt konsumiert werden. Die Anbieter der Kurse und die Betreiber der Plattform Ude-my teilen sich dabei die Gewinne: ein Kursanbieter erhält 97% der Gewinne nach allen Abzügen, wenn er einen Kursteilnehmer über einen Affiliatelink auf die Plattform bringt, während er 37% erhält, wenn dies nicht nachvollzogen werden kann [Ude22]. Das ist etwa dann der Fall, wenn jemand direkt über die Suche auf der Plattform einen Kurs kauft. Mit Abzügen beziehen sich dabei nicht nur auf die Mehrwertsteuer des jeweiligen Landes, sondern auch auf die Gebühr in Höhe von 30% des Gesamtpreises, welche an Apple und Google bezahlt werden muss, wenn ein Kauf direkt über die jeweilige native App getätigt wird.

4.2.2. Tests in Onlinekursen

Die Lernplattform Coursera erlaubt zusätzliche automatische Tests [vgl. Cou22a], welche am Ende der jeweiligen Lektionen bestanden werden sollen, um zur nächsten Lektion fortzufahren. Dabei handelt es sich um automatisierte Multiple Choice Tests oder Kurzantwort-Quizfragen. Die pädagogische Fraglichkeit von Multiple Choice Aufgaben wurde bereits in Kapitel 2.3 beschrieben, aber auch die Lösung der Kurzantwort-Quizfragen bringt Probleme mit sich. Unterscheiden sich die angegebenen Antworten in Wortwahl, Formulierung oder Grammatik, so kann diese nicht der korrekten Antwort zugeordnet werden und gilt somit als falsche Antwort. Durch diese Einschränkungen unterbietet die Usability der Kurzantwort-Quizfragen die pädagogische Sinnhaftigkeit. Ein Blogger kommentiert diese Tests mit: „So wie ich es verstanden habe, kannst du ein Quiz so oft wiederholen, bis du das Quiz bestanden hast. Manchmal musst du nach ein paar Versuchen jedoch eine gewisse Zeit warten, bevor du es erneut versuchen darfst. Die Lösungen von einigen Quizen sind sogar schon im Internet zu finden“ [Gei22]. Auch diese Aussage weist auf eine eher fragliche Bedeutung des Bestehens eines solchen Kurses hin.

4.2.3. Ergänzung um Lightbulb Learning

Die zweite Option für ein Geschäftsmodell für Lightbulb Learning besteht darin, den Autoren von Kursen zusätzlich zu ihren Einkünften auf den jeweiligen Plattformen Einkünfte

durch die Vergabe von Zertifikaten auf Lightbulb Learning zu ermöglichen. Kursteilnehmer können einem vom Kursautor verwalteten Kurs kostenlos beitreten und ihre offenen Fragen, Antworten und gegenseitiges Feedback beitragen. Die Mechanismen des Crowdsourcing sorgen dabei für gegenseitige Auslöser und Moderation. Der Evaluationsmechanismus, der alle Beiträge eines Kursteilnehmers übersichtlich, kontextualisiert und chronologisch sortiert darstellt, unterstützt den Kursautor bei der Vergabe von Fortschritt in Form einer Prozentzahl auf Grundlage der geleisteten Beiträge. Erreicht ein Teilnehmer den Schwellwert der 100%, so kann er zu einem vom Kursautor definierten Preis ein Zertifikat erhalten, welches in Form eines PDFs und als URL zur Verfügung gestellt wird. Der Erlös wird dann zwischen der Plattform und dem Kursautoren aufgeteilt. Dabei erhält der Kursautor einen höheren Prozentsatz als beispielsweise bei Udemy, so dass es eine wirtschaftlich kluge Entscheidung ist, aus Sicht eines Kursautors möglichst hochbepreisten Zertifikate auszugeben. Hypothetisch existiert aus Sicht der Kursautoren durch dieses Modell eine kausal zusammenhängende Kette von Mehrwerten, welche in den folgenden Abschnitten erörtert wird.

4.2.4. Mehrwert I: Steigerung der Lernqualität

Da die Komplexität, Adaptivität und Tiefe der Informationen, die man durch die Verwendung von Lightbulb Learning über das Thema des Onlinekurses diskutiert, größer ist, als wenn man die Informationen nur konsumiert, kann ein besseres Verständnis des Themas aufgebaut werden. Dies kann sich sowohl in der Leistung als auch in der Zufriedenheit der Teilnehmer, also sowohl objektiv als auch subjektiv, widerspiegeln. Da dies aus Sicht der Teilnehmer genau der Definition von Qualität des Kurses entspricht, folgen den Erfahrungen der Teilnehmer und deren Umfelds Weiterempfehlungen, welchen wiederum aus Anbietersicht eine Reichweitensteigerung folgt. Zusammenfassend lässt sich sagen, dass durch den Einsatz von Lightbulb Learning die Lernqualität eines Onlinekurses gesteigert werden kann.

4.2.5. Mehrwert II: Evaluationssystem und Zertifikate

Neben dem erhöhten Lerneffekt für die Teilnehmer und der daraus folgenden gesteigerten Qualität eines Kurses soll Lightbulb Learning ein System für die Evaluation der Teilnehmer ermöglichen. Besonders aktive Teilnehmer können von weniger aktiven Teilnehmern unterschieden werden, was eine quantitative Aussagekraft über die Leistungen der Teilnehmer impliziert. Zusätzlich kann der Kursautor sehr schnell einen qualitativen Eindruck gewinnen, indem die Beiträge jedes Teilnehmers chronologisch sortiert im

Kontext des Dialogs dargestellt werden. Auf dieser Grundlage kann er bestimmten Teilnehmern ein Zertifikat ausstellen, welches die Kenntnis in dem betreffenden Bereich attestiert. Diese Funktion könnte perspektivisch sogar direkt von Lightbulb Learning angeboten werden, so dass der damit verbundene Aufwand aus Sicht des Anbieters minimiert wird. Das Zertifikat hat für beide Beteiligten einen besonderen Anreiz: Für den Teilnehmer bedeutet ein Zertifikat, dass er Gelerntes nicht nur anwenden, sondern auch nachweisen kann. Dies kann beispielsweise Fortschritte in der Karriere, die Abhebung von Konkurrenz, fachliche Umstiege oder Kundenzuwachs ermöglichen. Dem Autoren eines Onlinekurses bietet sich durch das Anbieten von Zertifikaten ebenfalls Mehrwerte unterschiedlicher Art. Erstens ist die Herausgabe eines Zertifikats ein Verkaufsargument, mit dem er sich und seinen Onlinekurs von einem anderen Kurs zu dem gleichen Thema abheben kann. Ein zweiter Vorteil ist die indirekte Folge von der Herausgabe von Zertifikaten an richtig evaluierte Teilnehmer, da diese durch ihr Zertifikat für den Kurs werben. Durch die Filterung qualifizierter Teilnehmer erhält das Zertifikat auf lange Sicht eine Wirkung, welche, wie bei einer starken Marke, für etwas wertvolles steht und somit dem Herausgeber des Zertifikats bei der Bildung eines guten Rufs nützt.

4.2.6. Mehrwert III: Zusätzliche Einnahmequelle

Der wohl wichtigste Mehrwert und das stärkste Verkaufsargument für Anbieter von Online-Kursen ist die Absicherung der wirtschaftlichen Nachhaltigkeit. Mit Lightbulb Learning hat ein Anbieter die Möglichkeit, im Rahmen eines Affiliate-Programms, eine zusätzliche Einnahmequelle aufzubauen. Diese funktioniert so: Wird ein Kurs erstellt, der ab einer Evaluation von 100% ein Zertifikat verspricht, so kann der Kursautor einstellen, wie viel dieses Zertifikat kosten soll. Die Wahl des Preises obliegt dem Kursanbieter. Der Bezahlvorgang wird von Lightbulb Learning abgewickelt, und der Kursanbieter erhält einen prozentualen Anteil des eingestellten Preises, beispielsweise 80%. Somit kann der Anbieter seinen Teilnehmern gegenüber die Möglichkeit, ein Zertifikat zu erhalten, bewerben, sein Einkommen dadurch langfristig steigern, und die Abhängigkeit von der Preispolitik seiner aktuell verwendeten Lernplattform unabhängiger machen.

4.2.7. Marketing und Vertrieb

Das Geschäftsmodell B ist aus Perspektive von Lightbulb Learning auch deshalb interessant, weil die Eindringung in den Markt durch diesen Ansatz vereinfacht wird. Die zahlenden Kunden sind die Kursteilnehmer, so dass die Kursautoren ihr jeweiliges Pu-

blikum für die Teilnahme an den Kursen und den Erwerb der Zertifikate anwerben und dafür auch entsprechend incentiviert werden. Gegenüber den Kursautoren hingegen muss nicht verkauft werden, dass sie etwas für die Verwendung von Lightbulb Learning bezahlen müssen, sondern lediglich glaubhaft gemacht werden, dass die Verwendung eine zusätzliche Einnahmequelle und die beschriebenen zusätzlichen Mehrwerte bietet.

5

Serverless-Technologie

In diesem Kapitel geht es um eine Beschreibung und Abgrenzung der Grundbegriffe Serverless-Technologie und Reaktives System, sowie eine Einordnung der Zusammenhänge dieser Domänen.

5.1. Was ist Serverless-Technologie?

Serverless ist laut Red Hat ein „cloudnatives Entwicklungsmodell, bei dem Entwickler Anwendungen erstellen und ausführen können, ohne Server verwalten zu müssen“ [Red17]. Hier wird deutlich, dass der Begriff Serverless die Perspektive von Konsumenten und nicht von Anbietern dieser Technologieform beschreibt, da im physikalischen Sinne dennoch Server eingesetzt und verwaltet werden. Durch die Abstraktion generischer Problemfelder wird jedoch der Zugang dazu, insbesondere im Kontext der in der Vergangenheit etablierten On-Premise Lösungen, auf eine zielgerichtete Weise geschmälert und seitens des Betreibers mehr Verantwortung übernommen. Dies ist aus Entwicklersicht wertschöpfend, da man sich durch die Verwendung von Serverless-Technologie auf die Entwicklung von Geschäftslogik fokussieren kann, während Themen wie Betriebssysteme, Backups, Laufzeitumgebungen und Skalierung ausgelagert wird. Der Begriff ist nicht scharf genug definiert, um zu beinhalten, ab welcher Abstraktionsschicht eine Dienstleistung als Serverless bezeichnet werden darf, so definiert Red Hat beispielsweise auch Container-as-a-Service als Serverless [Red17], während umgangssprachlich häufig eher höhere Abstraktionsstufen wie z.B. Function-as-a-Service damit assoziiert werden.

5.2. Frontend im Kontext von Serverless-Technologie

Der Begriff von Serverless-Technologie bezieht sich nicht direkt auf die Implementierung des Frontends, welches auf dem Gerät des Entanwenders ausgeführt wird (z.B. als App oder als Web-Applikation in einem Browser). Dennoch kann sich der Stil der Frontendentwicklung dadurch tangiert werden, dass Geschäftslogik, welche keine sicherheitskritischen Informationen beinhaltet, in einem solchen Kontext eher direkt im Frontend abgebildet wird. Dies erfordert unter Umständen komplexere Strukturen und den bewussten Einsatz von Architekturen und Entwurfsmustern innerhalb des Frontends.

5.3. Function-as-a-Service (FaaS)

Mit dem Begriff der Function-as-a-Service ist gemeint, dass der Betrieb einer Laufzeitumgebung für die Ausführung einer Programmfunktion auf generische Weise von einem Cloudanbieter zur Verfügung gestellt wird. Ein Entwickler kann diese Dienstleistung nutzen, indem er seinen Code zum Cloudanbieter hochlädt, und dafür eine eindeutige Adresse erhält, anhand welcher diese Funktion zu finden und auszuführen ist. Die Funktion ist dabei frei von Seiteneffekten, welche das Dateisystem, das Betriebssystem oder die Laufzeitumgebung betreffen, kann aber beispielsweise Berechnungen, Datenbankabfragen oder Anfragen an andere Funktionen durchführen. Ergibt sich durch gehäufte Anfragelast eine Überforderung des Servers, auf welchem die Funktion ausgeführt wird, so sorgt der Cloudanbieter automatisch für eine elastische Skalierung. Das Besondere dabei ist, dass dabei durchaus auch auf 0 skaliert werden kann, was bedeutet, dass der Container oder die Laufzeitumgebung, in welcher die Funktion ausgeführt wird, gestoppt wird. In dem Fall wird dann bei Wiederausführung erst die benötigte Infrastruktur gestartet, und daraufhin die Funktion ausgeführt. Dies wird als cold start bezeichnet. Kosten werden in Abhängigkeit von der Anzahl der Aufrufe geltend gemacht, wobei häufig üppige kostenfreie Kontingente zur Verfügung stehen. Dadurch ermöglicht FaaS das kostengünstige Ausprobieren von Features, ohne initiale Investitionen in das Aufsetzen von Servern zu erfordern.

5.4. Reaktive Systeme

5.4.1. Antwortbereitschaft

Reaktive Systeme verfolgen das Ziel, zu jeder Zeit und gleichzeitig für beliebig viele Nutzer antwortbereit zu sein. Dafür werden drei weitere Qualitäten vorgeschrieben: Widerstandsfähigkeit, Elastizität sowie Nachrichtenorientiertheit, um die Antwortbereitschaft herzustellen [vgl. [Bon14](#)]. Diese Begriffe sowie deren Zusammenhänge sollen im Folgenden erklärt werden, um einen Vergleich zwischen den Prinzipien dieser beiden Paradigmen zu ermöglichen.

5.4.2. Widerstandsfähigkeit

Mit Widerstandsfähigkeit ist gemeint, dass das System „selbst bei Ausfällen von Hard- oder Software antwortbereit“ [[Bon14](#)] bleibt. Erreicht wird dies durch Replikation, Delegation und Isolation. Die Replikation der Funktionalität, wie beispielsweise durch mehrere Instanzen einer Komponente, kann Fehler einzelner Komponenten auffangen, indem die Anfragelast auf die funktionierenden Komponenten umgeleitet wird. Die Delegation von Verantwortung greift beispielsweise dann, wenn ein Fehlerfall auftritt, dessen Handhabung nicht von der Komponente selbst erreicht werden kann. In dem Fall kann sie die Abarbeitung an andere Komponenten delegieren und so selbst wieder in einen funktionalen Zustand zurückkehren. Auch die Isolation zwischen den Komponenten ist ein wichtiger Aspekt für die Widerstandsfähigkeit eines Systems, da sie dafür sorgt, dass an einer Stelle auftretende Fehler sich nicht auf andere Komponenten auswirken kann. Durch hohe Isolation bleibt der Schaden, den ein System durch einen Hardware- oder Softwarefehler nehmen kann, auf einen definierten Bereich eingeschränkt.

5.4.3. Elastizität

Ein elastisches System ist eines, welches in Abhängigkeit von der momentanen Anfragelast den Umfang der genutzten Ressourcen automatisch dynamisch festlegt. Sich ändernde Anfragelasten sind etwas sehr übliches in der Softwarebranche, Beispiele sind Abhängigkeiten von der Tages- oder Jahreszeit, bestimmten Veranstaltungen oder Hypes. Ein Beispiel ist der angekündigte Release eines Spiels, bei dem es die Möglichkeit zur Vorbestellung gab. Ab dem Zeitpunkt der Veröffentlichung wird dann schlagartig von sehr vielen Spielern mit hohen Erwartungen das Spiel heruntergela-

den, und es ist keine Seltenheit, dass die Server des Spieleherstellers in dem Moment Symptome der Überlastung zeigen¹. Das bedeutet konkret, dass bei einer sehr hohen Anfragelast zusätzliche Ressourcen, etwa in Form von Servern oder Kubernetes Nodes, zusätzlich gestartet werden, und die Anfragelast entsprechend aufgeteilt wird. Fällt die Anfragelast, so können diese zusätzlichen Instanzen automatisch auch wieder aufgelöst werden, um die dafür anfallenden Kosten zu sparen. Damit ein System überhaupt elastisch sein kann, muss es jedoch erst einmal skalierbar sein, wofür seinerseits Vorbedingungen gelten. Eine dieser Vorbedingungen ist die Nachrichtenorientiertheit.

5.4.4. Nachrichtenorientiertheit

In der Welt der verteilten Systeme muss ein Weg gefunden werden, die einzelnen Komponenten zuverlässig und eindeutig miteinander kommunizieren zu lassen. Für reaktive Systeme bedeutet das, dass die Nachrichten zwischen den einzelnen Komponenten asynchron und entkoppelt sind. Eine konkrete Umsetzung einer solchen Architektur lässt sich beispielsweise mit Apache Kafka oder dem Closed Source Equivalent Simple Notification Service der AWS erreichen. Es werden Topics definiert, auf welche Nachrichten gesendet werden können. Dabei ist dem Sender der Nachricht nicht wichtig, welche Komponenten die versendete Nachricht entgegennehmen. Auf den Topics werden die Nachrichten persistiert, während Komponenten auf diese reagieren können. Der wichtige Aspekt dabei ist die Entkopplung zwischen dem Sender der Nachricht und dem Empfänger. Durch die Überwachung der Nachrichten als Queue kann sowohl das Auftreten von Fehlern beobachtet werden (eine Komponente arbeitet keine Nachrichten mehr ab), als auch geschlussfolgert werden, ob mehr oder weniger Ressourcen als zum aktuellen Zeitpunkt zur Verfügung stehen benötigt werden. Somit ermöglicht die Nachrichtenorientiertheit sowohl die Widerstandsfähigkeit als auch die Elastizität, während diese beiden Eigenschaften antwortbereite und somit reaktive Systeme ermöglichen (vgl. Abbildung 5.1).

¹z.B. hier: <https://www.pcgamer.com/halo-infinite-steam-servers-slow/> und hier: <https://arstechnica.com/gaming/2021/11/rockstar-servers-down-for-nearly-a-day-amid-remastered-gta-trilogy-launch/>.

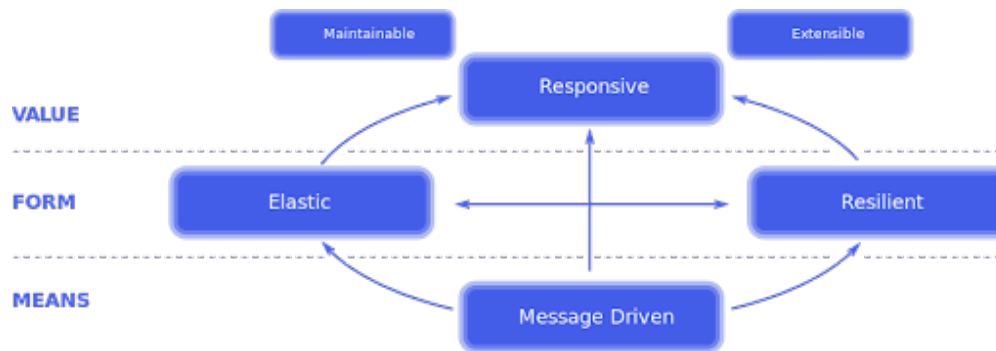


Abbildung 5.1: Die Darstellung der Abhängigkeiten der Qualitäten eines reaktiven Systems, [Bon14]

5.5. Reaktive Systeme und Serverless-Systeme

5.5.1. Gemeinsames Ziel: Antwortbereitschaft

Die Zielstellung von reaktiven Systemen und von Serverless-Prinzipien haben einige Gemeinsamkeiten, unterscheiden sich aber vor allem in der Perspektive auf diese wünschenswerten Eigenschaften. Um diese Gemeinsamkeiten und Unterschiede herauszuarbeiten, soll ein Blick auf die generelle Wertschöpfungskette in der Erstellung von Software geworfen werden. Aus wirtschaftlicher Sicht verfolgen die meisten Softwaresysteme das Ziel, einen Mehrwert durch Problemlösung des Nutzers zu schaffen. Dies kann ein System nur dann erreichen, wenn es auch entsprechend zuverlässig, schnell und korrekt antwortet, sobald ein Nutzer eine Anfrage macht. Dieses Ziel ist entsprechend unabhängig von den beiden Paradigmen, kann aber auch als primäres Ziel von beiden verstanden werden. Der Fokus von reaktiven Systemen liegt dabei allerdings eher auf der Struktur von Serveranwendungen, während die von Serverless-Technologie durch Vereinfachung beschleunigtes Nutzerfeedback und somit fachlich treffendere Softwareprodukte im Blick hat.

5.5.2. Delegation

Zur Erreichung des Ziels der Antwortbereitschaft wird, in beiden Paradigmen, die Delegation bemüht. Für reaktive Systeme bedeutet dies die Übertragung von Verantwortung von einer Komponente zu einer anderen, während beim Serverless-Paradigma eher die Übertragung der Verantwortung für den Betrieb von technischer Infrastruktur gemeint ist. Auch wenn sich die Bedeutung des selben Begriffs in den beiden Paradigmen sehr

stark unterscheidet, gibt es auch eine wichtige Gemeinsamkeit: Die Verantwortung wird nicht an einem einzigen Ort gebündelt, sondern durchdacht und bewusst auf unterschiedliche Instanzen aufgeteilt. Cloudanbieter wie AWS² oder Software-as-a-Service Firmen wie Supabase³, können wirtschaftlich und wertschöpfend agieren, indem sie mittels der Abstraktion von technischer Verantwortung viele Kunden mit der gleichen Lösung bedienen können.

5.5.3. FaaS und Elastizität

FaaS ist ein gutes Beispiel für eine elastische Serverless-Technologie. Wird die Anzahl der Anfragen schlagartig erhöht, so werden in einem FaaS Umfeld automatisiert und ohne Eingreifen des Entwicklers der Funktion oder des Cloudanbieters die Anzahl der Instanzen, welche die Anfragen bearbeiten können, erhöht. Wird hingegen die Anzahl der Anfragen daraufhin wieder reduziert, so werden die Instanzen von dem Cloud Anbieter auch wieder heruntergefahren, um die Hardware Ressourcen für andere Zwecke nutzen zu können. Dies geht so weit, dass bei längeren Zeitfenstern ohne Anfragen sogar alle Instanzen abgeschaltet werden. Es lässt sich festhalten: Die Verantwortung für die in reaktiven Systemen beschriebene Elastizität kann in Serverless-Systemen an die Cloudanbieter übertragen werden, so dass die damit einhergehende Komplexität nicht beim Entwickler der Funktion liegen, und gleichzeitig die daraus resultierenden Vorteile der Ressourcenoptimierung erzielt werden können. Aus Entwicklersicht spiegelt sich diese Elastizität also direkt in den Ausführungskosten der Funktionen wieder.

²<https://aws.amazon.com/de/>

³<https://supabase.com/>

6

Umsetzung: Lightbulb Learning

Die Umsetzung von Lightbulb Learning erforderte Aktivitäten unterschiedlicher Art. An dieser Stelle werden insbesondere die Erforschung der Lebenswelten der Nutzergruppe und die technologischen Entscheidungen beschrieben.

6.1. Interviews

6.1.1. Nutzerinterviews Studenten

Für Lightbulb Learning wurden eine Reihe von Interviews geführt, insbesondere mit den Zielgruppen Professoren und Studenten. Die Interviews wurden dabei nach den Leitprinzipien des Mom Test, wie sie im Kapitel 3.2 beschrieben wurden, strukturiert. Auch wenn sich die Fragen nicht in jedem Gespräch exakt glichen, sondern die Gespräche teils ihrerseits eine Entwicklung durchschritten, erwiesen sich die folgenden Fragen für die Zielgruppe der Studenten insbesondere als hilfreich:

1. Wie lernst du während des Semesters?
2. Wie lernst du (im Kontrast dazu) kurz vor der Prüfung?
3. Wenn du dich entscheiden müsstest: Ein gutes Ergebnis oder etwas Wichtiges gelernt?
4. Machst du nach einer Prüfung noch etwas für das Fach?

5. Wie würdest du lernen, wenn es keine Prüfungen mehr geben würde?

Diese Fragen wurden in Interviews mit 12 Studenten unterschiedlicher Studiengänge und Semester auf dem Campus und in der Mensa der HTWG Konstanz geführt. Die ersten zwei Fragen deuten den Kontrast zwischen dem langfristigen und dem kurzfristigen Lernen an. Es kam dabei heraus, dass der Großteil der Studenten explizit für die Prüfungen lernen, dabei das Ziel der Erreichung einer guten Note (oder das Bestehen selbst) im Kopf haben, und höchstens einige Wochen vor der Prüfung mit diesem Modus starten. Bei der Frage nach der Motivation für das Lernen antworteten etwa die Hälfte der Gesprächspartner, dass es ihnen wichtiger sei, etwas Wichtiges zu lernen, während die andere Hälfte sagte, dass die Note bzw. der Abschluss wichtiger sei. Diejenigen, die sich selbst als intrinsisch motiviert bezeichnet hatten, äußerten daraufhin mehrfach das Gefühl der Ertapptheit, da ihre Antworten auf die ersten beiden Fragen eher andeuteten, dass sie explizit für die Prüfungen lernen, und sich dabei weniger Aufwand während des Semesters machten. Eine weitere Beobachtung bei dieser Frage war, dass es sich aus Sicht der Studenten um eine schwierige Frage handelte. Dafür spricht einerseits die Unentschlossenheit und Bedenkzeit, welche die Studenten zeigten, als auch die Uneindeutigkeit im Vergleich zwischen allen Studenten. Eine deutlich höhere Eindeutigkeit fand sich in den Antworten auf die vierte Frage wieder, welche sowohl darauf abzielte, die Langfristigkeit des Lernwillens zu bezeugen oder zu widerlegen, als auch die Lebenswelt der Studenten in den Kontexten der Parallelität und Sequenzialität zwischen mehreren Modulen besser zu verstehen. Das Ergebnis: keiner der Gesprächspartner hatte sich bisher nach Prüfungen noch weiter mit den Inhalten der Vorlesung beschäftigt, bis auf eine Ausnahme, in der jemand eine Vorlesung als Nachschlagewerk benutzt hatte, als er in einem späteren Semester im Rahmen eines Projekts vor einem konkreten Problem stand. Die letzte Frage wurde bewusst offen bzw. hypothetisch formuliert, und bricht somit bewusst mit den Prinzipien des Mom Test. Das Ziel des Gedankenexperiments war die Teilhabe an der Vorstellungskraft der Studenten und die Einordnung der Prüfung auf Grundlage des Kontrasts, nämlich deren Abwesenheit. Die impulsive Reaktion der meisten Studenten deutete eine Befreiung von einem echten Problem oder Schmerz an, während nach etwas Bedenkzeit und bei genauerer Betrachtung häufig angefügt wurde, dass es dadurch an Antrieb, Fairness und Validität eines Abschlusses mangeln würde. Ein anderer Aspekt, der an dieser Stelle genannt wurde, war die Beschreibung teils bereits gängiger und guter Alternativen, wie etwa Projekte oder regelmäßige Abgaben, welche sich, aus Sicht einiger Gesprächspartner, als langfristige Lernmethoden erwiesen.

6.1.2. Nutzerinterviews Professoren

Da die Absicht Lightbulb Learnings unweigerlich mit dem Willen der Verantwortlichen es auch zu nutzen einhergeht, wurden 6 Professoren mit den folgenden Fragen interviewt.

1. Welches Ziel verfolgen Sie mit Ihren Prüfungen?
2. Wie Zufrieden sind Sie mit der Erreichung dieses Ziels?
3. Wie liefen die Online-Klausuren während der Pandemie?
4. Ist Ihnen wichtig, dass die Prüfung auch “die schlechten” Studenten herausfiltert?
5. Mit welchen anderen Prüfungsmethoden als der klassischen Klausur haben Sie Erfahrungen gemacht, und wie lauten diese?

Die Antworten auf die erste Frage lassen sich folgendermaßen zusammenfassen: Eine Prüfung stellt eine Form der Wissensabfrage mit dem Ziel der Evaluation dar. Es geht dabei nicht nur darum herauszufinden, ob bestimmte Inhalte wiedergegeben werden können, sondern auch darum, Gelerntes auf bestimmte Anwendungsfälle transferieren zu können. Bei Frage 2 wurde keine generelle Unzufriedenheit geäußert, jedoch berichteten einige Professoren, dass sie alternative Prüfungsmethoden wie mündliche Prüfungen, semesterbegleitende Projekte, Vorträge und regelmäßige Abgaben anwenden. Im Kontext der Pandemie (Frage 3) war die Meinung bis auf eine Ausnahme recht klar: Online-Klausur wurden als Problem angesehen, die größten Probleme waren die Abhängigkeiten von der technischen Infrastruktur und dem Umgang bei dessen Ausfällen und die fehlende Möglichkeit, die Anwendung von nicht zugelassenen Hilfsmitteln auszuschließen. Die Antworten auf die Frage 4 vielen eher zögerlich aus und gingen zum größten Teil in die Richtung, dass man eigentlich gerne allen Studenten dabei helfen würde, die Inhalte eines Kurses zu verstehen, als diejenigen, denen das nicht gelingt, herauszufiltern. Bei der letzten Frage wurden insbesondere die bereits genannten alternativen Prüfungsmethoden genannt, sowie eine Einschätzung in den Fragen der Effektivität und Machbarkeit in unterschiedlichen Kontexten.

6.1.3. Ergebnisse der Interviews

Die durchgeführten Interviews fanden nicht alle innerhalb einer kurzen Zeitspanne und auch nicht unter Laborbedingungen statt. Dadurch können die beschriebenen Zusammenfassungen in keiner Weise als quantifizierbare Erkenntnisse oder gar wissenschaftliche Datenerhebung genutzt. Dennoch konnte dadurch ein deutlich klareres Bild der

Perspektiven, Lebenswelten und Handlungsmotivationen der Zielgruppe erstellt werden. Diese Einblicke bestätigten das zu lösende Problem und konnten in die Produktentwicklung von Lightbulb Learning einfließen.

6.2. Technologie

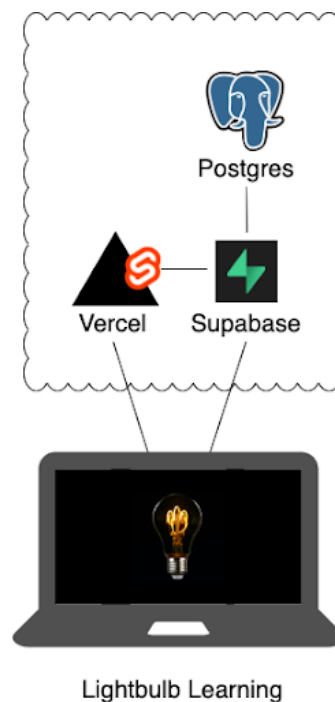


Abbildung 6.1: High-level Architekturübersicht Lightbulb Learning

In Abbildung 6.1 werden die grundlegend verwendeten Technologien dargestellt. Die Endnutzer greifen mittels eines Browsers auf die Server von Vercel zu, welche das Frontend ausliefern und anhand von SSR einen Teil der Anfragen direkt im Server durchführen. Dazu gehören auch Zugriffe auf die Datenbank, welche als PostgREST API¹ von Supabase bereitgestellt wird. Supabase betreibt und verwaltet für die Datenerhaltung eine Instanz einer PostgreSQL-Datenbank². Die folgenden Abschnitte gehen im Detail auf die Schnittstellen und Verantwortungsbereiche dieser Technologien ein.

¹<https://postgrest.org/en/stable/>

²<https://www.postgresql.org/>

6.3. Supabase

Supabase spielt in der Entwicklung von Lightbulb Learning eine zentrale Rolle, kann sogar als Schlüsseltechnologie bezeichnet werden. Eine Beschreibung der Aufgaben, die Supabase direkt und durch die Bereitstellung einer PostgreSQL Datenbank übernimmt, wird in den folgenden Abschnitten beschrieben.

6.3.1. Vergleich: Supabase und Firebase

Supabase beschreibt sich selbst als Open-Source Alternative zu Firebase³, welche „die Features von Firebase mithilfe von Enterprise-tauglichen Open-Source Tools“[Sup22b] nachbaut. Diese Funktionalitäten sind, neben einem zugänglichen Dashboard, im Wesentlichen die Bereitstellung einer Datenbank hinter einer im Internet verfügbaren Schnittstelle (bei Bedarf mit Echtzeit Subscriptions). Zusätzlich wird die Verwaltung der Authentifizierung und Autorisierung für die Zugriffe auf diese API und auf die dahinterliegenden Daten abgebildet. Auch die Möglichkeit, Dateien in einem Bucket abzulegen und Funktionen, welche auf dem Datenbankserver ausgeführt werden, wird zur Verfügung gestellt.. Dabei geht Supabase einige Aspekte auch anders als die eigens genannte Inspirationsquelle Firebase an, um weitere Mehrwerte neben dem Open-Source Gedanken anbieten zu können. Anstelle der dokumentenorientierten Datenstruktur in Firebase wählt Supabase ein relationales Datenmodell. Ein weiterer großer Unterschied ist die Art und Weise, wie auf die Daten zugegriffen wird. In Firebase gibt es dazu eine Frontend-Bibliothek, welche diese Aufgabe übernimmt, dabei allerdings sowohl intransparenter als auch weniger flexibel als der Client in Supabase ist. Während die Frontend-Bibliothek von Supabase im Wesentlichen lediglich eine JavaScript-Repräsentation der Abfragesprache übernimmt und diese den entsprechenden REST Aufrufe zuweist, ist bei Firebase Firestore der präferierte Weg vom Browser zur Datenbank immer den offiziellen Client zu verwenden. Dieser Client behält sich vor, die Datenstruktur sowie auch die Schnittstelle in Richtung der Datenhaltung zu verändern, so dass hier keine eigene Implementation für diese Aufgabe begünstigt wird. Die bestehende Implementierung ist dabei gleichzeitig nicht für den Einsatz auf der Serverseite ausgelegt, so dass die Entwicklung einer Applikation mit Server-Side Rendering deutlich erschwert wird. Für die Entwicklung nativer Applikationen werden entsprechend Bibliotheken bereitgestellt, um auch dies zu ermöglichen. Aufgrund dieser inhaltlichen Unterschiede zeigt sich, dass Supabase nicht nur eine Firebase Alternative ist, sondern lediglich ein Teil der gleichartigen Probleme adressiert. Gleichzeitig

³<https://firebase.google.com/>

ist die Parallele zwischen den beiden Technologien aus der Perspektive des Marketings für Supabase ein sehr guter Ansatz, um Entwicklern und potentiellen Kunden, denen Firebase häufig schon ein Begriff ist, sehr schnell zu vermitteln, welches Problem Supabase löst.

6.3.2. Relationale Datenbanken

Welche Eigenschaften eine relationale Datenbank für den Anwendungsfall einer Applikation mit strukturierten Daten nützlich macht, soll in diesem Abschnitt diskutiert werden. Grundsätzlich ist die Idee einer relationalen Datenstruktur die Verbindung von Zeilen unterschiedlicher Tabellen anhand von Schlüsseln, welche eindeutige Zuweisungen ermöglichen. Dadurch kann eine Abfragesprache genutzt werden, welche die Zusammenhänge zwischen mehreren Tabellen beachtet oder diese zur Abfragezeit nach Bedarf miteinander kombiniert. Die relationale Datenbank garantiert dabei die Eigenschaften der Atomarität, Konsistenz, Isolation und Dauerhaftigkeit von Änderungen, dies wird üblicherweise mit dem Akronym ACID abgekürzt. Atomarität bedeutet, dass eine Transaktion stets entweder komplett durchgeführt wird, oder, im Falle des Scheiterns einer Transaktion, der ursprüngliche Zustand vollständig wiederhergestellt wird. Die Aufrechterhaltung der Konsistenz bedeutet, dass die Relationen zwischen den einzelnen Tabellen zu jedem Zeitpunkt entsprechend der formulierten Regeln (und somit intakt) gehalten wird. Die Isolation ist auf der zeitlichen Achse zu betrachten: Ist eine Transaktion noch nicht beendet, so können die Änderungen, die innerhalb dieser Transaktion stattfinden, nicht von einer parallel stattfindenden Transaktion gelesen werden. Die Änderungen sind somit so lange voneinander isoliert, bis eine Transaktion erfolgreich abgeschlossen wurde. Die Dauerhaftigkeit von Daten beschreibt, dass die Daten einer erfolgreichen Transaktion auf einer Festplatte persistiert werden. Wird das System beispielsweise von der Stromversorgung getrennt und zu einem späteren Zeitpunkt wieder angeschlossen, so sind die vorher gespeicherten Daten immer noch verfügbar.

6.3.3. PostgreSQL (alias Postgres)

Supabase betreibt eine PostgreSQL bzw. Postgres Datenbank (beide Bezeichnungen werden austauschbar verwendet [Pag07]). PostgreSQL ist eine etablierte relationale Datenbank, welche seit dem ersten öffentlichen Release 1996 aktiv von der Open Source Community weiterentwickelt wird. Der Fokus dabei liegt auf der Zuverlässigkeit, Datenintegrität, Erweiterbarkeit, Performanz und Skalierbarkeit [vgl. The22]. Gleichzei-

tig verfügt PostgreSQL über eine Vielzahl an Features⁴, von denen einige im Kontext der Serverless-Anwendungsentwicklung sich als besonders nützlich erweisen. Dazu zählen insbesondere die PostgreSQL Row-Level-Security Regeln (RLS) und die PostgreSQL Functions, welche in den folgenden Abschnitten erläutert werden.

6.3.4. Row-Level-Security

Die Row-Level-Security Funktionalität von PostgreSQL⁵ kann für die Definition von Zugriffsregeln auf die Daten genutzt werden. In Abbildung 6.2 ist ein Beispiel aus Lightbulb Learning für den lesenden Zugriff auf die Tabelle `course_user`. Die Definition der Regel ist seinerseits eine SQL-Abfrage, welche einen Wahrheitswert, welcher über die Blockierung der weiteren Durchführung entscheidet, zurückgibt.

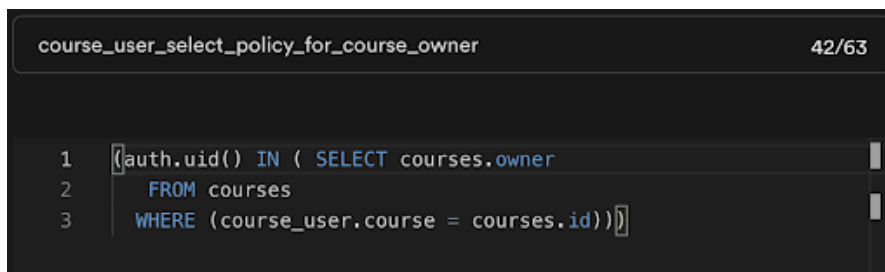


Abbildung 6.2: Row-Level-Security-Regel Beispiel für Kurs-Besitzer

In diesem Beispiel ist das SQL Statement wahr, falls der anfragende User als Besitzer des Kurses gelistet ist. Für die Teilnehmer des Kurses ist eine weitere Regel hinterlegt, da sich die Anfragen inhaltlich unterscheiden. Die Verwendung von Row-Level-Security Regeln erweist sich als gangbare Alternative zur Entwicklung eines eigenen Backends zur Verwaltung der Zugriffsrechte auf Zeilenebene der Datenbank, da diese Komplexität eingespart werden kann, gleichzeitig aber die Feinheit der Granularität der Zugriffsregeln beliebig gesteigert werden kann. Für noch komplexere Regeln oder die Wiederverwendung von Strukturen können auch PostgreSQL Functions genutzt werden. Diese werden im Folgenden Abschnitt erläutert.

6.3.5. PostgreSQL Functions

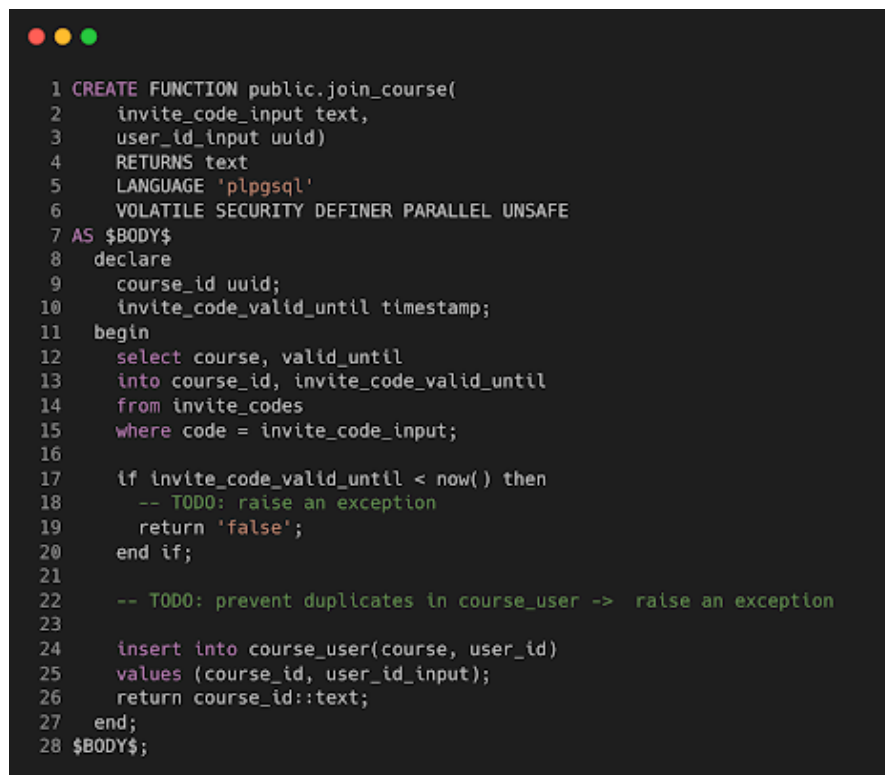
Das Feature von PostgreSQL Functions⁶ bietet die Möglichkeit, serverseitigen Code auszuführen, ohne dabei die Laufzeitumgebung oder andere zusätzliche infrastrukt-

⁴<https://www.postgresql.org/about/featurematrix/>

⁵<https://www.postgresql.org/docs/current/sql-createpolicy.html>

⁶<https://www.postgresql.org/docs/current/sql-createfunction.htm>

turelle Aspekte neben der Datenbank selbst verwalten zu müssen. Da der Code direkt auf der Datenbank ausgeführt wird, können die definierten Rollen (im Sinne der Autorisierung) verwendet werden, um sicherheitsrelevante Abfragen zu machen. Dies ist beispielsweise dann von Belang, wenn private Token verwendet werden, welche nicht an den Client ausgeliefert werden dürfen. Ein zweiter Anwendungsfall ist die Wiederverwendung von Logik in unterschiedlichen Kontexten, etwa bei der Definition von Row-Level-Security Regeln. Ein weiterer Anwendungsfall ist die Einbeziehung von äußeren Faktoren in eben solche Regeln, welche nicht in Form von einer SQL Abfrage abbilden lassen. Ein Beispiel für diesen Fall findet sich in der folgenden Abbildung 6.3.



```
1 CREATE FUNCTION public.join_course(  
2     invite_code_input text,  
3     user_id_input uuid)  
4     RETURNS text  
5     LANGUAGE 'plpgsql'  
6     VOLATILE SECURITY DEFINER PARALLEL UNSAFE  
7 AS $BODY$  
8 declare  
9     course_id uuid;  
10    invite_code_valid_until timestamp;  
11 begin  
12     select course, valid_until  
13     into course_id, invite_code_valid_until  
14     from invite_codes  
15     where code = invite_code_input;  
16  
17     if invite_code_valid_until < now() then  
18         -- TODO: raise an exception  
19         return 'false';  
20     end if;  
21  
22     -- TODO: prevent duplicates in course_user -> raise an exception  
23  
24     insert into course_user(course, user_id)  
25     values (course_id, user_id_input);  
26     return course_id::text;  
27 end;  
28 $BODY$;
```

Abbildung 6.3: Die join_course Funktion

Zuerst fällt die etwas umständliche und unflexible Definition der Funktion auf, welche der verwendeten Sprache PL/pgSQL geschuldet ist. Dies steht für Procedural Language/Postgres Structured Query Language und erweitert SQL um unter anderem um Va-

riablen, Bedingungen und Schleifen in Form einer prozeduralen Sprache⁷, welche bereits 1998 erschienen ist. Für einfachere Fälle ist es auch möglich, PostgreSQL Functions in SQL zu schreiben, falls diese Spracherweiterungen nicht benötigt werden. Im Beispiel wird in den Zeilen 1-6 der Kopf der Funktion selbst definiert. Dazu gehören der Name, die Eingabeparameter, die verwendete Sprache und der Rückgabewert. Die Zeilen 8-10 definieren die verwendeten Variablen, welche im eigentlichen Rumpf der Funktion, also in den Zeilen 12-26, verwendet werden dürfen. Hier wird zu einem bestimmten Einladungscode der Ablaufzeitpunkt mit dem aktuellen Zeitpunkt verglichen. Ist der Einladungscode noch nicht abgelaufen, so wird der Nutzer dem entsprechenden Kurs hinzugefügt und der Primärschlüssel des Kurses zurückgegeben. Da dieses Beispiel sowohl die Möglichkeiten dieser prozeduralen Spracherweiterung, als auch die damit einhergehenden Schwierigkeiten besonders plakativ darstellt, wird im Kapitel 7 dazu detaillierter Stellung bezogen. Auch wenn Seiteneffekte in Form von Zustandsänderungen der Datenbanktabellen ein natürlicher Effekt von PostgreSQL Functions sind, so kann doch durch die Anwendung von Transaktionalität auf SQL-Ebene der Umgang mit Fehlern eindeutig geregelt werden, beispielsweise durch die Durchführung eines Rollbacks in diesem Fall. Nachdem eine Funktion definiert ist, wird sie von Supabase in der API zur Verfügung gestellt [vgl. Sup22a], und kann vom Client aus direkt aufgerufen werden.

6.3.6. PostgREST

PostgREST ist eine Serveranwendung, welche eine generische Zuordnung zwischen den relationalen Datenstrukturen und -operationen der PostgreSQL-Datenbank und dem [Fie00, Kapitel 5] vornimmt. So kann beispielsweise für das Löschen einer Zeile in der Datenbank ein REST DELETE genutzt werden. Dabei wird eine Referenz auf eine Ressource (aus REST Sicht), welche dem Primärschlüssel der Zeile in der PostgreSQL Datenbank entspricht, zur Identifikation des zu löschenden Elements angegeben. In Abbildung 6.4 wird beispielsweise das Löschen eines Kurses durch einen HTTP Aufruf dargestellt. Die Anfragemethode ist in diesem Beispiel DELETE, während die Id, also der Primärschlüssel des Kurses, als Anfrageparameter übergeben wird.

⁷<https://www.postgresql.org/docs/current/plpgsql-structure.html>

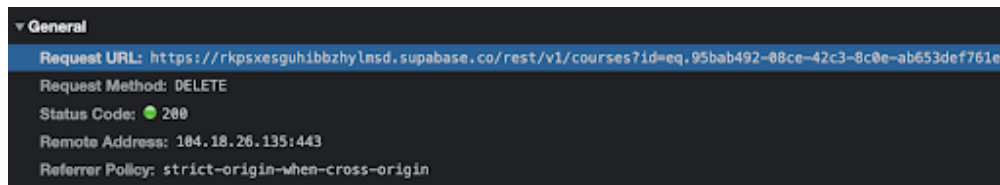


Abbildung 6.4: Screenshot der Netzwerküberwachung in Chrome beim Löschen eines Kurses

PostgREST identifiziert diesen API-Aufruf als Löschanfrage, übersetzt es in die entsprechende SQL Anfrage und führt diese auf der PostgreSQL-Datenbank aus. Ist die SQL Anfrage erfolgreich, so wird wiederum der entsprechende Status Code 200 zurückgegeben. Das gleiche Muster funktioniert ebenfalls für das Anlegen, Auslesen und Verändern von Elementen mit den jeweiligen REST und SQL Äquivalenten. PostgREST ermöglicht mittels der Definition von Konfliktlösungs- und Verschmelzungsstrategien sowie der Kombination von Anfrageparametern die Möglichkeit, auch komplexere Anfragen durchzuführen, ohne dafür eine serverseitige Applikation schreiben, testen, betreiben und skalieren zu müssen. Damit fügt sich die Kombination aus PostgREST und PostgreSQL mit seinen Funktionen wie Row-Level-Security und Functions, als gehostete Dienstleistung von Supabase, in den Komplex einer Serverless-Anwendung ein.

6.4. Web Applikation mit SvelteKit

Entsprechend der Anforderung der hohen Entwicklungsgeschwindigkeit und größtmöglichen Flexibilität fiel die Entscheidung zwischen den zur Verfügung stehenden Frontend-Frameworks wie beispielsweise VueJS⁸, Angular⁹ und React¹⁰ auf ein sehr modernes Framework namens SvelteKit¹¹.

6.4.1. SvelteKit und Svelte

SvelteKit wird auf der eigenen Homepage als Framework für Webapplikationen jeder Größe, mit einem flexiblen Dateisystem-basierten Routing beschrieben. Es funktioniert so, dass man, ähnlich wie bei den übrigen verbreiteten Frameworks auch, Komponenten als Dateien in einem eigenen Format beschreibt. Streng genommen muss man dabei zwischen SvelteKit und Svelte¹² unterscheiden. Svelte ist das von SvelteKit genutz-

⁸<https://vuejs.org/>

⁹<https://angular.io/>

¹⁰<https://reactjs.org/>

¹¹<https://kit.svelte.dev/>

¹²<https://svelte.dev/>

te Komponenten-Framework, dessen Aufgabe es ist, aus Interaktivität, Struktur und Styling bestehende Komponenten zur Kompilierungszeit in hoch optimiertes JavaScript mit HTML und CSS zu kompilieren. Dies gelingt über die Ausnutzung der labelled Syntax in der Sprachdefinition von JavaScript, durch welche dynamische Abhängigkeiten zwischen Variablen ausgedrückt werden können [vgl. [Sve22](#)], was wiederum den Verzicht auf das Virtual DOM ermöglicht [vgl. [Har18](#)]. Neben der Nutzung optimierter Svelte-Komponenten folgt SvelteKit einem weiteren, sehr mächtigem Prinzip, nämlich dem der Transitional Apps. Um dieses zu erklären, ist es wichtig, sich die deutlich verbreiteteren Begriffe der Multi-Page Application (MPA) und der Single-Page Application (SPA) vor Augen zu führen.

6.4.2. Multi-Page Apps (MPAs)

Multi-Page Applications sind der traditionelle Weg der Auslieferung von Web-Applikationen. Dabei werden Routen auf Ressourcen gemappt, welche mit voneinander separaten Aufrufen vom Browser aus abgefragt und dargestellt werden. Dieser Vorgang kann auch durch Server-Side Rendering (SSR) unterstützt werden, welches die ausgelieferte HTML-Datei anhand von Anwendungslogik auf dem Server zusammensetzt. Dieses Vorgehen hat einige Vorteile. Wird eine Ressource aufgerufen, so kann diese ohne weitere Zwischenschritte direkt und schnell, optional sogar ohne JavaScript-Anteile, ausgeliefert werden. Die Anwendungslogik kann im Server abgebildet werden, so dass Entwickler eine größere Auswahl an Technologien zur Verfügung haben. Des Weiteren können die Features des Browsers, wie beispielsweise die Navigationselemente, so genutzt werden, wie sie ursprünglich gedacht waren: Zum Navigieren zwischen den Seiten. Es bestehen allerdings auch einige Nachteile: Zwischen den einzelnen Aufrufen kann kein Zustand geteilt werden, was bedeutet, dass beispielsweise bei der Verwendung von JavaScript-Bibliotheken sämtlicher Code neu geladen und evaluiert werden muss. Dies verlangsamt die Navigation innerhalb einer Web-Applikationen spürbar, auch deshalb, weil jede neue Seite einzeln über das Netzwerk übertragen werden muss. Auch für Designprinzipien wie das der Objektpermanenz sind Multi-Page Apps schlichtweg nicht ausgelegt, so dass visuelle Transitionen zwischen zwei unterschiedlichen Seiten unmöglich werden.

6.4.3. Single-Page Apps (SPAs)

Der modernere Ansatz der Single-page Applikationen löst einige dieser Probleme, erzeugt dabei aber zugleich einige neue Nachteile. Die Grundidee von SPAs ist, dass

sämtliche für die Darstellung der Applikation notwendigen Aspekte beim initialen Laden der Webseite heruntergeladen und gecached werden. Durch die Benutzer-Interaktion wird das DOM direkt im Browser manipuliert, ohne dabei eine weitere Seite mit den üblichen Browser-Mechanismen zu laden. Dadurch wirken SPAs, nach einer etwas verlängerten initialen Ladezeit, sehr schnell und weisen daher beispielsweise bei der Web-Applikation internen Navigation Ähnlichkeiten mit lokal installierten Applikationen auf. Allerdings stößt man durch die Anwendung dieses Prinzips auch sowohl auf Grenzen, etwa beim Laden dynamischer Inhalte welche unmittelbar mit Eingaben des Nutzers zusammenhängen, als auch auf konkrete Nachteile, wie die erhöhte initiale Ladezeit beim Aufruf einer Webseite. Zusätzlich wird das Routing in einer SPA virtualisiert, womit eine gewisse Fehleranfälligkeit einhergeht. Außerdem bedeutet es, dass das Öffnen eines Deeplinks dazu führt, dass die gesamte Applikation anstelle der eigentlich angefragten Ressource geladen werden muss. Innerhalb dieser Applikation muss dann die ursprünglich angefragte URL als Eingabeparameter gedeutet werden, und die entsprechenden Inhalte müssen dargestellt werden. Dadurch wird die Umsetzung von Deeplinks nicht nur deutlich verlangsamt, sondern auch zu einem expliziten, optionalen und fehleranfälligem Feature einer SPA, es ist entsprechend also denkbar, dass eine Web-Applikation Deeplinks überhaupt nicht unterstützt.

6.4.4. Transitional Apps

SvelteKit befolgt das Prinzip der Transitional Apps [vgl. [Har21](#)]. Dieser Begriff ist noch sehr jung und wenig in den einschlägigen Referenzen verankert, beschreibt aber treffend den Umgang mit den Abwägungen, die zwischen MPAs und SPAs gemacht werden müssen. Es soll widerspiegelt werden, dass die Vorteile beider Prinzipien, sowohl der traditionellen, als auch der modernen, kombiniert werden. Ursprünglich stammt der Begriff aus dem Interior Design und wird als „blend of traditional styles with the contemporary styles of the day“ [[Wea22](#)], also einer Kombination aus traditionellem und zeitgenössischem Stil beschrieben. Die Übertragung dieser Begrifflichkeit und dem dahinter liegenden Prinzip bedeutet: Eine transitionale App ist einerseits vorwärtsgewandt und hat einen Blick auf die Vorteile der Lösungen, die eine SPA anbieten, respektiert aber gleichzeitig die Vorteile der traditionellen MPAs, und versucht insgesamt, die Vorteile beider Prinzipien miteinander zu kombinieren [vgl. [Har21](#)].

6.4.5. Adapter, Server-Side Rendering und Hydration

SvelteKit erreicht dieses Ziel unter anderem durch sogenannte Adapter¹³, welche für SvelteKit Projekte konfigurierbar sind. Die Aufgabe eines Adapters ist die Umwandlung einer kompilierten SvelteKit-Applikation in ein Deployment-Paket, welches für eine bestimmte Deployment-Umgebung ausgelegt ist. Dadurch kann der Einsatz von Server-Side Rendering lose von unterschiedlichen Deployment-Providern und den jeweils verwendeten Laufzeitumgebungen gewährleistet werden. Beispiele für Deployment-Provider, für welche solche Adapter bereits existieren, sind Vercel¹⁴, Netlify¹⁵ und Cloudflare¹⁶. Durch das Server-Side Rendering können Komponenten, welche nur aus statischen Anteilen bestehen, vorgerendert und dadurch ohne weitere Zwischenschritte direkt ausgeliefert werden können. Komponenten mit dynamischen Aspekten werden zur initialen Anfragezeit vorerst als statisches HTML-Dokument ohne Interaktivität provisioniert. Gleich im Anschluss daran werden benötigte dynamische Anteile nachgeladen, so dass die Seite nicht nur dargestellt, sondern auch damit interagiert werden kann. Dieser Prozess der Hydration¹⁷ ist für den Nutzer nicht sichtbar, führt aber in seiner Wahrnehmung zu sehr kurzen initialen Ladezeiten. Im Anschluss lädt SvelteKit die Seiten vor, welche von der aktuellen Seite aus direkt erreicht werden können. Dies führt dazu, dass bei der Navigation zu anderen Seiten innerhalb der Transitional App der interne Router von SvelteKit die Steuerung der Browser-Navigation übernehmen kann. Dadurch kann auf die zusätzliche Fehleranfälligkeit, welche durch die Umsetzung dieser Funktionalität durch den App-Entwickler selbst entstehen würde, verzichtet werden. In diesem Kontext werden Frameworks, welche diese Aufgaben übernehmen, werden auch Meta-Frameworks genannt. Alternativen wären beispielsweise Nuxt¹⁸ für Vue und NextJS¹⁹ für React.

6.5. TypeScript

Für die Entwicklung der Applikation wurde insbesondere TypeScript²⁰ verwendet. TypeScript ist eine Erweiterung von JavaScript, welche von Microsoft weiterentwickelt und

¹³<https://kit.svelte.dev/docs/adapters>

¹⁴<https://vercel.com/>

¹⁵<https://www.netlify.com/>

¹⁶<https://www.cloudflare.com/>

¹⁷<https://kit.svelte.dev/docs/page-options#hydrate>

¹⁸<https://nuxtjs.org/>

¹⁹<https://nextjs.org/>

²⁰<https://www.typescriptlang.org/>

als JavaScript mit der Syntax für Typen beworben wird. Die stark typisierte Sprache bietet einen umfassenden Satz an Werkzeugen für die Entwicklungsumgebung, welche die Entwicklung beschleunigen und die Fehlerwahrscheinlichkeit zur Laufzeit verringern, indem es Typen eindeutig macht und Fehler bereits zur Kompilierzeit aufdeckt. Dadurch erhöht sich auch die Testbarkeit der Applikation, da durch die Typisierung nach der Kompilierung mehr über die Objekte bekannt ist, und somit die Anzahl der möglichen Fälle bekannt. Da Browser lediglich JavaScript Code ausführen können, wird der TypeScript Code zu optimiertem JavaScript transpiliert. Da sowohl TypeScript, als auch die etablierte Entwicklungsumgebung Visual Studio Code als open source Projekte von Microsoft verwaltet werden, sind diese gut miteinander integriert.

6.6. Skalierung der Datenbank

Für die Analyse der Antwortbereitschaft von Lightbulb Learning ist die Datenbank als einzige zentrale Instanz als Flaschenhals für die Skalierung im Erfolgsfall zu sehen. Daher stellt sich die Frage, ob man eine PostgreSQL-Datenbank so skalieren kann, dass für alle Kunden eine positive Benutzererfahrung erhalten bleibt. Darauf hat die PostgreSQL-Datenbank und dessen Community einige Antworten, welche sowohl im Bereich der vertikalen, als auch in der horizontalen Skalierung liegen. Die vertikale Skalierung entspricht der Erhöhung der Ressourcen für eine einzelne Instanz. Für die Benutzung durch Supabase entspräche das dem Upgrade von der Free auf die Pro Version, wobei hier auch teilweise Elemente der horizontalen Skalierung enthalten sind). Als relevantere Frage ergibt sich somit: Was, wenn die horizontale Skalierfähigkeit von PostgreSQL durch Supabase limitiert wird? In dem Fall könnten die Daten mit einem einfachen Befehl aus Supabase extrahiert werden, und eine eigene Datenbank-Infrastruktur aufgebaut werden. In dieser könnten dann alle Konzepte der horizontalen Skalierung angewendet werden, welche auf der Ebene von PostgreSQL ermöglicht werden. Für Applikationen, welche überwiegend lesende Zugriffe auf die Datenbank machen, wie das bei Lightbulb Learning der Fall ist, wird empfohlen, weitere Replicas hinzuzufügen, auf welche die Lesezugriffe aufgeteilt werden können. Würde die Applikation mit der Menge an Schreibzugriffen an ein Limit stoßen, so kann die Datenbank auf mehrere Server partitioniert werden, indem entweder unterschiedliche Tabellen auf unterschiedlichen Datenbankservern gehalten werden, oder sogar eine einzige Tabelle auf mehrere Server mittels Sharding aufgeteilt wird. Dennoch gibt es auch hier eine Grenze, welche durch die Erhaltung der Konsistenz Garantie entsteht. Die Wahrscheinlichkeit des Eintretens dieser Bedarfe ist zwar sehr gering, aber den-

noch ist es allgemein eine sinnvolle Absicherung, im Erfolgsfall einen Plan zu haben, die technische Infrastruktur mit sprunghafter Nachfrage skalieren zu können. Supabase in Kombination mit PostgreSQL ermöglichen die Umsetzung dieses möglichen Bedarfs.

6.7. Testing mit Playwright

Für das Testen von Lightbulb Learning wird Playwright²¹ verwendet, einem von Microsoft betreuten open source Werkzeug zur Testautomatisierung, welches bei der Initialisierung von SvelteKit Projekten mit dem CLI Tool als Standard Testing-Framework vorgeschlagen wird. Mit Playwright lassen sich Anwendungsfälle durch die direkte Interaktion mit dem Browser testen und debuggen, die Tests entsprechen also echten Ende-zu-Ende Tests. Dadurch können die Tests gegen eine Produktivumgebung ausgeführt werden. Die Möglichkeit, die Tests mit unterschiedlichen Browsern auszuführen, so dass auch hier Bugs aus Nutzersicht früh entdeckt und gelöst werden können, erhöht verringert zusätzlich die Wahrscheinlichkeit für blinde Flecken. Im Gegensatz zu Cypress²², einem konkurrierenden und zu Beginn des Projekts verwendeten Testframework, können durch die Verwendung der modernen async/await Strukturen und automatischen Retries resiliente und aussagekräftige Tests geschrieben werden, die sich auf unterschiedlich schnellen Umgebungen gleich Resultate liefern. Das Besondere an diesen Tests ist aber, dass sie die Funktionen des Systems aus Nutzersicht beschreiben, und als lebende Dokumentation dessen betrachtet werden, was das System zum Zeitpunkt der Ausführung wirklich zu tun vermag. Ist ein bestimmter Test erfolgreich, so ist damit unter Voraussetzung korrekter Implementierung des Tests validiert, dass das System eine gewünschte Eigenschaft erfüllt. Schlägen Tests fehl, so ist das Gegenteil der Fall, und es muss etwas dafür getan werden, dass die gewünschten Eigenschaften wieder erfüllt werden. Dieses Prinzip wurde bereits 2006 im Artikel über Behavior-Driven Development [vgl. [Nor06](#)] (BDD) von Dan North erläutert.

6.8. Continuous Deployment Pipeline mit GitHub Actions

Für die Kompilierung, automatische Testausführung und automatische Deployments wurde eine Continuous Deployment Pipeline mit GitHub Actions²³ aufgebaut, welche

²¹<https://playwright.dev/>

²²<https://www.cypress.io/>

²³<https://github.com/features/actions>

durch Interaktion mit der Versionsverwaltung ausgelöst wird. Jeder Commit führt zur automatischen Durchführung aller Tests (unabhängig vom Branch), dafür wird auf dem GitHub Actions Server eine lokale Datenbank gestartet und mit Testdaten initialisiert, der Entwicklungsserver gestartet, der das Frontend ausliefert und im headless Modus die definierten Anwendungsfälle getestet. Bei GitHub Actions handelt es sich ebenfalls um eine Serverless-Technologie. Schlägt mindestens ein Test fehl, wird eine Mail mit einem Fehlerprotokoll versendet. Unabhängig vom Testergebnis wird daraufhin ein Preview-Deployment auf Vercel durchgeführt, so dass jede Version der Applikation bei Bedarf unter realistischen Bedingungen manuell getestet werden kann. Die einzige Ausnahme dafür sind Commits, welche auf den main branch gepusht werden: Diese werden automatisiert auf die Produktivumgebung deployed.

7

Evaluation

In diesem Kapitel geht es um eine Evaluation der Idee, der verwendeten technologischen Prinzipien und deren Zusammenhänge zum methodischen Kontext. Dabei werden sowohl die Technologien, welche nun aktuell im Produkt verwendet werden, als auch jene, welche verwendet aber im Anschluss wieder ausgebaut wurden, beschrieben.

7.1. Experiment: Reaktive Systeme

Für den Kurs Reaktive Systeme wurde Lightbulb Learning für die Prüfungsvorbereitung der mündlichen Prüfung verwendet. Der Kurs bestand zu dem Zeitpunkt der Prüfung aus 7 Teilnehmern, welche in einem Zeitraum von 3 Wochen die Kursinhalte in Lightbulb Learning diskutieren konnten. Für die mündliche Prüfung wurden dann einige dieser Fragen verwendet, so dass die Prüflinge durch die Mitgestaltung der Fragen und Antworten und das Lernen dieser Fragen aktiv ihre Prüfungsergebnisse verbessern konnten. Dabei entstanden insgesamt 26 offene Fragen und 42 Antworten. Das Funktion zum Geben von Feedback wurde nicht genutzt. Eine Übersicht der Beiträge der Prüflinge ist in Abbildung 7.1 abgebildet.

Prüfling	Anzahl der Fragen	Anzahl der Antworten	Summe der Beiträge	% Anteil an Beiträgen gesamt
1	9	16	25	36,76%
2	7	11	18	26,47%
3	5	7	12	17,65%
4	5	4	9	13,24%
5	0	3	3	4,41%
6	0	1	1	1,47%
7	0	0	0	0,00%
Summe	26	42	68	100,00%

Abbildung 7.1: Darstellung der Anzahl der Beiträge gruppiert nach Prüfling

Es fällt auf, dass die Anzahl der Beiträge insgesamt nicht gleichverteilt sind, sondern wie in einer Pareto-Verteilung ein kleiner Teil der Studenten einen Großteil der Beiträge leistete und anders herum. Dies ist vermutlich durch die Freiwilligkeit bzw. die fehlende direkte Incentivierung bedingt, da die eigentliche Evaluation in der mündlichen Prüfung erfolgte. Etwa die Hälfte der Prüflinge zeigte durch die Involvierung in Diskussionen ihre intrinsische Motivation, und konnten diese Leistung zu ihrem Vorteil nutzen, wenn sie am Ende der Prüfung zwischen zwei Noten standen. In dem Fall wurde die Leistungsübersicht als weiterer Messpunkt für die Evaluation zurate gezogen. Schätzungsweise wäre das Ergebnis gleichverteilter ausgefallen, wenn der Lightbulb Learning Kurs als einziges Medium der Evaluation, wie beispielsweise bei Onlinekursen, genutzt werden würde. Dennoch gaben dieses Experiment bzw. die Teilnehmer des Kurses, auch neben den quantifizierbaren Größen geringer Messgröße, einige qualitative Hinweise auf die Benutzbarkeit der Software. Beispiele dafür waren das überflüssige Konzept der Entwürfe für Fragen, Antworten und Feedback, welches eher zur Verwirrung als zur Absicherung vor fälschlicherweise veröffentlichten Inhalten führte. Diese Funktion wurde daraufhin aus Lightbulb Learning herausgenommen. Ein anderer Einblick in die Perspektive mehrerer Benutzer war der Wunsch, gepostete Inhalte noch einmal anpassen zu können. Auch die Idee, bereits auf der Ebene der Frage die Anzahl der Antworten, und vielleicht sogar die richtige Antwort anzuzeigen, entstand in dieser Feedbackschleife. Auch der Login-Prozess, welcher den Zugriff auf die E-Mails auf dem jeweiligen Gerät voraussetzt, wurde kritisiert. Dieser Aspekt wurde durch die Integration des Logins mit GitHub als Authentifizierungsprovider adressiert. Somit verhalf dieses Experiment als erste breitere Anwendung des Systems und realistischen Bedingungen zu frühem, anwendbarem Feedback auf den Ebenen der Fachlichkeit und Technologie.

7.2. Experimente mit AWS Lambda und Scala 3

Während der Planung dieser Thesis war angedacht, in Scala 3 entwickelte AWS Lambda Funktionen für die Geschäftslogik zu verwenden. Dieses Vorgehen erwies sich als weniger effektiv als erwartet, so dass auch auf der Ebene der Technologieauswahl eine Kurskorrektur erfolgte. Die Gründe dafür sollen in diesem Abschnitt beschrieben werden.

7.2.1. Probleme mit AWS Amplify

AWS Amplify¹ ist eine Werkzeugsammlung von AWS, welche sich die schnelle Entwicklung von Fullstack-Apps mit der AWS Cloud zum Ziel gesetzt hat. Überraschenderweise erwiesen sich diese Werkzeuge jedoch, aus einer Reihe von Gründen, als ungeeignet. Der wichtigste Grund war die Verwendung der aws-amplify Bibliotheken in Kombination mit dem Bundling Tool Vite² für ES Modules, welches das Standardwerkzeug für das Bundling von SvelteKit-Applikationen ist. Die aws-amplify Bibliothek kann nicht mit Vite verwendet werden, da sie die veraltete CommonJS Struktur nutzt. Über einen konfigurativen Umweg mit manuellen Anpassungen an tieferliegenden technischen Schichten und dem Verzicht auf Vite war es dann möglich eine Svelte-App auf AWS Amplify zu deployen. Die Anforderung eines vollständigen Frontend-Frameworks, nämlich SvelteKit, konnte dadurch jedoch nicht mehr erfüllt werden. Diese Inkompatibilitäten werden regelmäßig in der Community diskutiert, wurden bisher aber noch nicht vom Provider aufgelöst. Hinzu kommt ein deutlich stärkeres Abhängigkeitsverhältnis zum Cloudprovider von AWS, der sogenannte Vendor Lock-in.

7.2.2. Lambda Kaltstarts bei JVM-basierten Funktionen

Als FaaS Dienst wurde AWS Lambda verwendet. Scala 3 Funktionen wurden als .JAR-Datei gebündelt und inklusive aller verwendeten Bibliotheken zu AWS hochgeladen. Ein Kaltstart tritt dabei immer dann auf, wenn der Container, in welchem die Funktion ausgeführt wird, vor der Ausführung erst gestartet werden muss. AWS macht keine genaue Angabe darüber, nach welcher Inaktivitätsdauer die Container heruntergefahren werden, empirische Erfahrungswerte schwanken in der Größenordnung einiger Minuten. Es gibt jedoch keine Garantie über die Dauer, über die ein Container mindestens aktiviert bleibt. Das bedeutet, dass jede Anfrage an eine Funktion einen Kaltstart darstellt, sofern nicht wenige Minuten vorher bereits eine Anfrage gemacht wurde. Der große

¹<https://aws.amazon.com/de/amplify/>

²<https://vitejs.dev/guide/why.html>

Nachteil von JVM-basierten Funktionen wie etwa solchen, die in Scala 3 geschrieben sind, ist die Länge der Kaltstarts. Diese beträgt für einfache fachliche Funktionen mit wenigen einbezogenen externen Bibliotheken zwischen 25 und 40 Sekunden. Es gibt einige Lösungen für dieses (vorher bereits bekannte) Problem. Beispiele hierfür wären die Nutzung von Architekturmustern, welche die Asynchronität begünstigen, wie Command Query Responsibility Segregation [vgl. [Fow11](#)], und Event Sourcing [vgl. [Fow05](#)] aus dem Umfeld des Domain-Driven Design [vgl. [Eva04](#)]. Hier wird durch eine strikte Trennung zwischen Lese- und Schreibzugriffen (CQRS) und dem Persistieren von Serien von Events anstelle der aktuellen Zustände von beispielsweise Aggregaten (Event Sourcing) eine Systemstruktur etabliert, welche prinzipiell gut mit den durch Asynchronität erzeugten Latenzen zurecht kommt. Ein Beispiel für einen Aufruf in einem CQRS System ist in Abbildung 7.2 dargestellt.

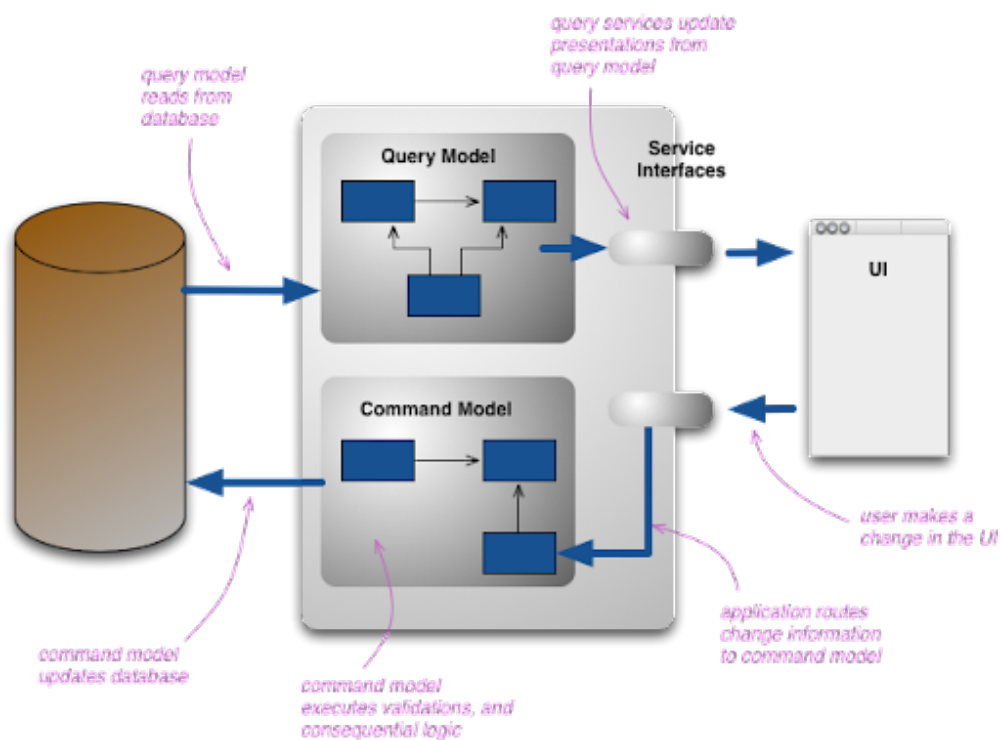


Abbildung 7.2: Konzeptionelle Übersicht über einen Aufruf an ein CQRS System, [[Fow11](#)]

Der Aufruf beginnt unten rechts mit einem Nutzer, welcher über eine Benutzeroberfläche eine Änderung im System macht. Die Änderung wird an einen Endpunkt weitergeleitet, welcher ausschließlich für Schreibzugriffe genutzt werden kann. Der Endpunkt leitet die angeforderte Änderung an das Command Modell weiter. Dieses kennt die Validierungsregeln des Aggregats und etwaige Implikationen und Abhängigkeiten, so dass die Anfrage diesen Regeln folgend bearbeitet kann. Ist eine Anfrage fachlich gültig, so wird die Änderung an die Datenbank propagiert, und der Schreibzugriff ist vollständig.

Der Lesezugriff wird davon getrennt aufgerufen, wofür ein Lesemodell (Query Model) benutzt wird. Die Struktur dieses Modells ist auf das Lesen optimiert, so dass hier effizient der aktuelle Zustand des Systems an den Benutzer zurückgegeben werden kann, selbst wenn ein Kaltstart darin enthalten ist. Der Kern dieser Idee beruht auf der Annahme, dass die meisten Applikationen deutlich häufiger Lese- als Schreibzugriffe durchlaufen, so dass eine Optimierung auf das Lesen mittels eines eigenen Lesemodells lohnenswert ist. Experimentell war es im Kontext von Lightbulb Learning grundsätzlich möglich, einige solcher eventbasierten Funktionen zu schreiben. Dennoch erwies es sich als sehr schwierig, im Zusammenspiel mehrerer Funktionen, ein System zu gestalten, welches den in Kapitel 3.3 beschriebenen Anforderungen an die Werkzeuge gerecht wird. Funktionen gehören per se keinen Gruppen zu, wie es beispielsweise Endpunkt zu Microservices tun, so dass eine strikte Aufrechterhaltung dieser Architekturprinzipien nicht intuitiv ist. Hinzu kommen die langen Kaltstarts, welche nicht mit den Standard Timeouts gängiger Browser von etwa 30 Sekunden vereinbar sind, sowie die damit verbundene katastrophale User Experience. Selbst wenn bei Schreibzugriffen das gewünschte Ergebnis zuerst angenommen werden kann, und im Fehlerfall durch eine verspätete Fehlermeldung dies korrigiert werden kann, so ist hierbei der Zusammenhang zwischen Handlung und Feedback so weit auseinander, dass eine Zuordnung für den Nutzer als schwierig gestaltet. Hinzu kommt, dass lesende Zugriffe zwar bereits in Aspekten der Datenstruktur und des zugehörigen Modells optimiert sind, eine Seite, die mehr als wenige Millisekunden lädt dabei allerdings nicht hinnehmbar ist. Somit kann man festhalten, dass die typischen eventbasierten Architekturmuster zwar durch ihre asynchrone Natur Latenzen abfedern können und für Systeme, in denen die Auslöser für Aufrufe keine Menschen sondern andere Systeme sind, geeignet sind. Für JVM basierte Lambdas reicht diese Federung für menschliche Nutzer allerdings schlichtweg nicht aus. Eine potentiell geeigneteres Architekturmuster für FaaS wird im folgenden Abschnitt beschrieben.

7.2.3. FaaS API ist keine Aktoren API

Während der Entwicklung des FaaS-basierten Ansatzes kam die Idee auf, dass Prinzipien wie sie im Aktorenmodell, bereits 1973 von Carl Hewitt [vgl. [Hew73](#)] definiert wurden, für die Architekturgestaltung für FaaS basierte Systeme sinnvoll sein könnten, da es einige interessante Parallelen gibt. Im übertragenen Sinne würde dann eine Instanz einer FaaS-Funktion einer Instanz eines Aktors entsprechen. Wird ein Aktor aufgerufen, so kann er im wesentlichen Nachrichten empfangen, Berechnungen durchführen, eine Nachricht an einen ihm bekannten Aktor versenden, neue Aktoren erzeugen oder festlegen, wie er auf einen nächsten Aufruf reagieren wird [vgl. [Hew10](#)]. Das Ziel die-

ser Struktur ist die Aufteilung der Anwendungslogik in kleinere, hierarchisch strukturierte und insbesondere leichtgewichtige Teilkomponenten mit eindeutig definierten Verantwortungsbereichen und Typen. Beispielsweise wirbt Akka³, einer Implementierung des Aktorenmodells von Lightbend⁴, mit ungefähr 2,5 Millionen Aktoren pro GB Heap-Speicher. Durch diese Eigenschaften eignen sich Aktoren als gute Grundlage für sehr stark skalierbare, reaktive Systeme (siehe Kapitel 5.4). Würde es gelingen, eine Übersetzung dieser Prinzipien für das FaaS-Äquivalent zu entwickeln, so könnte möglicherweise ein noch weitaus skalierbares System konstruiert werden. In diesem würde selbst das Aufsetzen der benötigten Infrastruktur außerhalb des Verantwortungsbereichs des Entwickler liegen. Das dafür erforderliche Framework könnte sich dabei hinsichtlich der API an der Aktoren API von Akka orientieren, wurde für Lightbulb Learning allerdings nicht tiefergehend verfolgt. Die Hauptgründe für diese Entscheidung war das fortwährende Problem der Kaltstarts sowie die (aktuelle) Schwergewichtigkeit von AWS Lambdas. Andere, leichtgewichtere FaaS Provider könnten sich in dieser Hinsicht als besonders durchsetzungsstark erweisen. Beispiele dafür sind Cloudflare Workers⁵ oder Deno Deploy⁶, welche durch andere Implementierung des FaaS Konzepts die Kaltstarts für ein vernachlässigbares Problem machen oder gar ganz lösen, während durch Deployments auf Edge Server [vgl. Meu18], die Zugriffszeiten noch weiter verringert werden. Damit ist gemeint, dass Funktionen nicht nur auf einem oder weniger definierten Rechenzentren installiert und ausgeführt werden, sondern stattdessen weltweit verteilt sind. Dadurch können Nutzer immer auf das jeweils nächstgelegene Rechenzentrum zugreifen und somit die Netzwerklatenzen deutlich verkürzen. Eine detailliertere Bezugnahme auf Deno Deploy folgt im Ausblick.

7.2.4. Libraries und Dokumentation für Java

Ein weiteres Problem bei der Entwicklung von Scala 3 Lambdas war das geringe Vorhandensein von Dokumentation. Es gibt zwar für alle wesentlichen Schritte von der Entwicklung einer solchen Funktion, über das Einbinden von externen Bibliotheken und automatisierte Bereitstellen Anleitungen und Referenzen. Allerdings sind diese nicht offiziell von AWS und werden entsprechend nicht zwangsläufig aktuell gehalten, erfüllen nicht zwangsläufig einen Mindeststandard an Qualität und sind nicht unbedingt korrekt. Nachdem es dennoch gelungen war, Scala 3 Funktionen zu schreiben, zu deployen und

³<https://akka.io/>

⁴Interessanterweise ist einer der Co-Autoren des Reaktiven Manifests, Jonas Bonér, gleichzeitig auch der CEO von Lightbend und ursprüngliche Entwickler von Akka.

⁵<https://workers.cloudflare.com/>

⁶<https://deno.com/deploy>

aufzurufen, war ein weiteres Problem die Dokumentation des AWS-SDKs für Java⁷. Die Übersetzung von Java Konzepten auf Scala 3 Konzepte muss man selbst übernehmen. Dies ist nicht nur mit viel Aufwand verbunden, sondern hat auch zur Folge, dass die erwartete Schlankheit von funktionalen Programmen durch Übersetzungen zwischen diesen zwei Konzepten leidet.

7.3. Svelte und SvelteKit

7.3.1. Intuitive Konzepte

Die Entwicklung der Lightbulb Learning Applikation als transitionale App, welche die Vorteile von SPAs und MPAs kombiniert, erwies sich insgesamt als sehr produktiv und zielführend, auch wenn es einige Stolpersteine zu überwinden gab. Auch wenn vor der Entwicklung von Lightbulb Learning mit diesen Frameworks kaum Erfahrung vorhanden war, so konnten sich die wichtigsten Konzepte durch gute Dokumentation und eine hilfsbereite Community schnell angeeignet werden. Dabei ist insbesondere zu bemerken, dass sowohl Svelte als auch das Metaframework SvelteKit einige strukturelle Konzepte anwenden, welche der Intuition des Entwicklers entsprechen, ohne dabei bewusst diese Konzepte gelernt zu haben. Bei Svelte gehört dazu beispielsweise die Kombination aller semantisch zu einer Komponente gehörenden Styling-, Struktur- und Funktionselemente in einer einzigen Datei. Bei SvelteKit erfüllt das Dateisystemgestützte Routing diesen Anspruch, welches Entwicklern aus der klassischen MPA-Entwicklung bereits vertraut ist.

7.3.2. Stolperstein: Serverseitige Authentifizierung

Der bereits angedeuteten Stolpersteine sollte man sich jedoch auch bewusst sein, wenn man die Entscheidung für die Verwendung dieses Frameworks trifft. Ein Problem, welches bei der Entwicklung einer SvelteKit Applikation immer wieder auftaucht, ist der Mangel an externen Referenzen wie Tutorials, Codebeispiele und Bibliotheken, besonders im Vergleich zu Referenzen der aktuell gängigen Frameworks wie React, Angular und Vue. Ein besonders stark spürbares Beispiel ist das Nichtvorhandensein einer Bibliothek für die Server-side Authentifizierung mit Supabase. Diese ist zwar für Next.js und Nuxt bereits vorhanden, für SvelteKit bisher allerdings lediglich angekündigt [vgl. Sch22]. Die aktuelle Alternative, eine eigene Implementierung, ist dabei hinsichtlich der

⁷<https://aws.amazon.com/de/sdk-for-java/>

hohen Komplexität und der damit verbundenen gelegentlichen Unzuverlässigkeit, nicht zufriedenstellend. Daher wird sie ersetzt, sobald eine offizielle Lösung veröffentlicht wird. Der aktuelle Ansatz basiert darauf, dass der Nutzer nach der initialen Authentifizierung das Access Token als Cookie bei jedem Request mitschickt, welches dann für die serverseitigen Prefetches wiederverwendet wird.

7.3.3. NodeJS Laufzeitumgebung ist kein Browser

SSR mit SvelteKit bedeutet, das Frontend-Code sowohl auf dem Server als auch auf dem Client ausgeführt wird. Auf dem Server wird dazu eine NodeJS-Umgebung verwendet. Allerdings unterscheiden sich diese Umgebung in mancherlei Hinsicht, so dass es sein kann, dass in den Code Weichen eingebaut werden müssen, welche das Verhalten abhängig von der jeweiligen Laufzeitumgebung beschreiben.

7.3.4. PostgreSQL as a Service mit Supabase und lokale Entwicklung

Die Entwicklung der Persistenzschicht mit Supabase erwies sich als pragmatisch, niederschwellig und zuverlässig. Dabei passte das Vorgehensmodell bei der Adaption dieses Dienstes gut zu den generellen Prinzipien bei diesem Produkt und der Serverless-Technologie allgemein. Zu Beginn wurde direkt auf der Webseite von Supabase ein Projekt angelegt und ein paar Tabellen generiert. Dabei konzentrierte sich der Lerneffekt insbesondere auf den Umgang mit der grafischen Benutzeroberfläche sowie einer Wiederauffrischung der Grundprinzipien des Umgangs mit PostgreSQL. Die erzeugten Tabellen waren vorerst vollständig öffentlich und es konnte ohne Authentifizierung auf diese zugegriffen werden. Daraufhin wurde die automatisiert provisioniert API vom Client aus konsumiert, um ein Gespür für den Umgang mit PostgREST und der inkludierten clientseitigen Bibliothek zu entwickeln. Anschließend folgte die Ersetzung des Login und Logout Features der Supabase Bibliothek, welche zur Reduktion der Komplexität vorerst ohne SSR die auf Amplify basierende Authentifizierung ersetzte. Nachdem es für Benutzer möglich war, sich zu Authentifizieren, war der nächste logische Schritt die Einbeziehung von RLS Regeln, welche den Zugang zu den Datensätzen hinter der API begrenzen. Mit diesem Vorgehen wurden noch einige weitere Schritte gemacht, bis hin zur Entwicklung mit einer lokalen Instanz von Supabase und der Versionsverwaltung der Datenbankstruktur in Form von Migrationsskripts, welche mittels der Supabase CLI angelegt, verwaltet und sogar auf die Produktivumgebung ausgerollt werden. Es zeigte sich, dass sich die Verwendung von Supabase aus der Perspektive eines Entwicklers sehr inkrementell gestalten lässt. Daher wird von einem frühen, funktionierenden

Stand ausgegangen, und dieser kann schrittweise um viele weitere technische Schichten ergänzt werden, ohne dabei zu große Lernschritte vorauszusetzen. Der Umgang mit Supabase erwies sich durch die Kleinschrittigkeit und den inkrementellen Aufbau der Lernkurve als angenehm und für Projekte dieser Art als empfehlenswert.

7.3.5. Serverless nur für CRUD-Szenarien?

Eine verbreitete Ansicht ist, dass Serverless sich lediglich für Standardfälle wie CRUD (Create-Read-Update-Delete) Szenarien eignen würde, während bei komplexeren Anforderungen stets mehr Verantwortung über die Infrastruktur selbst übernommen werden müsse. Diese Ansicht kann mit Lightbulb Learning zwar nicht widerlegt werden, aber es soll als gutes Gegenbeispiel dienen. Es wurde bewusst darauf geachtet, dass möglichst viele unterschiedliche Anforderungsarten in der Implementierung berücksichtigt wurden. Das Ergebnis ist, dass es zu jedem Problem, für welches es in traditionellen Weltsicht eine Lösung gibt, auch in den Sphären der Serverless-Technologie eine äquivalente Lösung gab. So konnten, ohne einen einzigen Backendservice, einfache und komplexere logische Zusammenhänge abgebildet werden. Daten können langfristig, zukunftsicher und ohne Vendor-Lockin persistiert werden, es gibt öffentliche und private Daten und von letzterer Art kann die Datensicherheit und -integrität gewährleistet werden. Auch das Hosten der Webseite, die Ausführung von Funktionen auf der Datenbank, die Build-Pipeline und die Versionsverwaltung des Quellcodes kommen ohne eigene infrastrukturelle Verantwortung aus. Der Querschnitt von Funktionen, welcher sich über mehrere technische, gestalterische und geschäftliche Schichten erstreckt, konnte von einem einzigen Entwickler vollständig in dem Zeitraum der Thesis umgesetzt werden. Lightbulb Learning wird nun von einigen Professoren und Studenten an der eigenen Hochschule verwendet, und es wurde Interesse an dem Produkt von einem wissenschaftlichen Mitarbeiter der Pädagogischen Hochschule Karlsruhe bekundet.

7.4. Serverless-Prinzipien

Drei Prinzipien, die alle Serverless-Technologien gemeinsam zu haben scheinen, sollen in diesem Absatz zusammengefasst werden. Diese lauten

1. Abstraktion
2. Delegation
3. Fokus

und stehen im folgenden Abhängigkeitsverhältnis zueinander. Technische Herausforderungen, wie beispielsweise der Betrieb einer Datenbank, werden abstrahiert. Das bedeutet beispielsweise, dass die Verwaltung der benötigten Infrastruktur automatisiert wird. Durch diese Abstraktion wird die Delegation möglich. So kann beispielsweise durch die zur Verfügungstellung einer entsprechenden Schnittstelle ein Nutzer die abstrahierte technologische Herausforderung als Dienstleistung nutzen, und dadurch Aufwand sparen. Zusätzlich können meta-Herausforderungen wie Skalierbarkeit und Elastizität in einer Dienstleistung mitinbegriffen sein. Das möglicherweise entstehende Abhängigkeitsverhältnis muss, insbesondere aus der Perspektive des Nutzers einer solchen Dienstleistung, bewusst gehandhabt werden. Gerade in einer frühen Produktphase, in welcher der Fokus auf der Evaluations einer Produktidee liegt, sind die Vermeidung solcher Abhängigkeiten mit hohem Aufwand verbunden. Durch die Delegation der genannten technischen Schichten erhöht sich somit der Fokus der Entwicklung auf das Geschäftsmodell. Dies hat Einfluss auf angrenzende Bereiche, wie in den Kapiteln 8.2 und 8.3 diskutiert wird: Ideen können früher evaluiert werden, und die Iterationszeit durch den Build-Measure-Learn Zyklus lässt sich verkürzen.

8

Fazit

In diesem Kapitel geht es darum, die Zusammenhänge zwischen den unterschiedlichen Bereichen herzustellen, und eine Einordnung in den Kontext der Frage zu machen, wie innovative digitale Produkte in der Zukunft entwickelt werden können. Zuletzt geht es um die Frage, ob es sich bei Lightbulb Learning um ein innovatives Produkt handelt.

8.1. Lightbulb Learning, ein reaktives System?

Das in reaktive Manifest beschreibt Prinzipien, welche zu robusteren, resilienteren, flexibleren und besser platzierten Systemen führen [vgl. [Bon14](#)]. Historisch ist dies in den Kontext der sich etablierenden Microservices-Architekturen, im Gegensatz zu den bis dahin üblichen, monolithisch-strukturierten Systemen, einzuordnen. Selbst frühe Cloud Anbieter wie AWS waren zu dem Zeitpunkt erst wenige Jahre alt und erfuhren starkes Wachstum, was als Alternative zu on-premise Systemen auf sogenannten bare metal Maschinen gesehen wurde. Auch die gesellschaftliche Einordnung ist wichtig: 2014 hatten Softwaresysteme durch die zunehmende Verfügbarkeit von Smartphones, Breitband-Internetanschlüssen und sich mit der Digitalisierung ändernde Geschäftsmodelle neue Anwendungsfälle, welche eine viel größere Skalierung, als es bisher nötig war, erforderten. Software musste immer und überall funktionieren, und bereits geringe Unterschiede in Latenzen oder Kompatibilitäten zu kleineren Geräten hatten wirtschaftlich weitreichende Folgen. Die im reaktiven Manifest beschriebenen Eigenschaften eines Systems beziehen sich auf diesen Kontext, der sich in den letzten 8 Jahren seinerseits weiterentwickelt hat. Mit Serverless-Technologie können viele der Probleme, die

man durch die Anwendung des reaktiven Manifests lösen kann, ausgelagert und somit umgangen werden. Natürlich gibt es dafür Ausnahmen, Beispiele könnten aus den Bereichen des Machine Learnings oder des Internet-of-Things kommen. Mit Lightbulb Learning konnte jedoch ein Beispiel geschaffen werden, mit welchem es möglich ist, ganz ohne eigene Verantwortung über die Infrastruktur ein stets antwortbereites und skalierbares System zu erstellen. Somit lässt sich zusammenfassen: Mit Lightbulb Learning werden nicht alle Eigenschaften reaktiver Systeme erfüllt, beispielsweise gibt es kein Konzept der Nachrichtenorientiertheit. Dennoch wurde das Ziel erfüllt, welches für reaktive Systeme definiert ist, nämlich durch Skalierbarkeit und Elastizität ein stets antwortbereites und resilientes System zu entwickeln.

8.2. Zusammenhang zwischen Lean Startup und Serverless-Technologie

Um ein Produkt nach den Lean Startup Prinzipien zu gründen, bedarf es nicht unbedingt der Serverless-Technologie. Dennoch lässt sich darin ein Fokus auf möglichst schlanke Experimente entdecken, welche durch den Einsatz dieser Gruppe an Technologien die Umsetzung solcher Experimente erleichtern. Die wesentliche Eigenschaft von Serverless-Technologie ist dafür die Delegation von technologischer bzw. infrastruktureller Verantwortung an außenstehende Dienstleister. Zusammenfassend kann man also sagen, dass Serverless-Technologie für innovative Produkte nach Lean Startup zwar nicht unbedingt benötigt wird, es aber die Umsetzung dieser Ansätze vereinfacht.

8.3. Zusammenhang zwischen agiler Entwicklung und Serverless-Technologie

Eine ähnliche Beziehung besteht auch zwischen der agilen Entwicklung und Serverless-Technologie. Auch wenn methodologische Prinzipien an sich keine bestimmten Technologien voraussetzen, so erkennt man doch einige Parallelen in den zugrundeliegenden Prinzipien, woraus man schließen kann, dass Serverless-Technologie den Einsatz von agilen Prinzipien begünstigt. Ein konkretes Beispiel dafür ist die Unterteilung, Priorisierung und der Fokus der Arbeit auf fachliche Blöcke, und nicht auf technische, was sich auf dem ersten agilen Prinzip stützt, dass die höchste Priorität die Zufriedenstellung des Kunden durch frühe und kontinuierliche Auslieferung wertvoller Software sei [Bec01].

Da rein technische Schichten ohne die Umsetzung der fachlichen Anforderungen des Kunden dies nicht erfüllen können, arbeitet man stattdessen entlang der gewünschten Funktionalität. Eine Frage, welche in der Anwendung dieses Vorgehens häufig entsteht, ist die Rolle von technischen Abhängigkeiten. Ein Beispiel könnte lauten, dass für die Umsetzung von Feature A und Feature B ein neuer Microservice aufgesetzt werden muss, was mit einem bestimmten infrastrukturellen Aufwand verbunden ist. Wenn nicht klar ist, ob zuerst Feature A oder erst Feature B umgesetzt wird, was eine der wesentlichen Eigenschaften agiler Entwicklung ist, so ist es beispielsweise schwer, eine valide Schätzung der Komplexität der Umsetzung der jeweiligen Funktionen abzugeben. Die Gruppe solcher Probleme wird durch den Einsatz von Serverless-Technologie vereinfacht, da der Aufwand, der in solche technischen Themen investiert werden muss, insgesamt verringert wird. Es lässt sich also schließen, dass über die reine Kompatibilität zwischen den beiden Ansätzen hinaus der Einsatz von Serverless-Technologie den von agiler Entwicklung vereinfacht.

8.4. Ist Serverless-Technologie die Zukunft der Apps?

Es konnte gezeigt werden, dass Serverless-Technologie einige Eigenschaften erfüllt, welche typische Herausforderungen der Applikationsentwicklung adressiert. Der Trend in der Entwicklung von Software zu immer höheren Graden der Abstraktion von Technologie, um den Fokus auf die Geschäftslogik zu schärfen, setzt sich mit Serverless-Technologie fort. Die wichtige Frage wird bleiben, ob Serverless-Technologie die wichtigsten Probleme, die es bei der Entwicklung von digitalen Produkten gibt, adressiert. Durch die in den Abschnitten 8.2 und 8.3 aufgezeigten Zusammenhänge lässt sich das zwar vermuten, da diese Frameworks in der nahen Vergangenheit einen großen Problem adressiert haben, allerdings könnten die großen Herausforderungen der Zukunft (in Hinsicht der Applikationsentwicklung) in dieser Frage auch aus Bereichen stammen, die zum gegenwärtigen Zeitpunkt nur schwer abgeschätzt werden können.

8.5. Lightbulb Learning, eine Innovation?

In diesem Abschnitt wird eruiert, ob es sich bei Lightbulb Learning tatsächlich um eine Innovation handelt. Der österreich-amerikanische Ökonom Joseph Alois Schumpeter definiert fünf Arten von Innovationen: Produktinnovationen, welche sich auf neuartige Produkte beziehen, Verfahrensinnovationen, welche durch neue Produktionsverfah-

ren effizientere oder bessere Produkte anbieten und Markttinnovationen, welche durch die Erschließung neuer Absatzmärkte neue Einkommensströme generieren. Außerdem werden Beschaffungsinnovationen beschrieben, welche einen Wettbewerbsvorteil durch die Erschließung neuer Quellen bieten, sowie Strukturinnovationen, welche durch Neustrukturierung der Arbeitsorganisation effizientere Abläufe die daraus entstehenden besseren Produkte oder Dienstleistungen beschreiben. Eine Voraussetzung für eine echte Innovation ist außerdem die daraus erwachsende wirtschaftliche Wertschöpfung, welche sie von einer Erfindung unterscheidet [vgl. [Sle13](#)]. Unter dieser Einschränkung kann Lightbulb Learning bisher lediglich den Anspruch erheben, eine potentielle Innovation zu sein, endgültig ist dies allerdings erst durch wirtschaftlichen Erfolg zu belegen. Dabei entspräche Lightbulb Learning einer Produktinnovation und insbesondere einer Verfahrensinnovation.

8.5.1. Produktinnovation

Die Idee hinter Lightbulb Learning, wie sie in Kapitel [2.6](#) beschrieben wird, ist unter der genannten Einschränkung eine Produktinnovation, da zwei Eigenschaften von Leistungsbewertung im Bereich der Lernevaluation miteinander kombiniert werden konnten: Formativität und Skalierbarkeit.

8.5.2. Verfahrensinnovation

Um diese Produktinnovation zu implementieren, wurden innovative Werkzeuge aus dem Bereich der Serverless-Technologie verwendet. Diese ermöglichen eine Umsetzung von einer einzelnen Person in einem verhältnismäßig kurzen Zeitraum und somit einer deutlichen Geringhaltung der Kosten, was wiederum eine frühe Validierung ermöglicht. Ein Blick auf das gleiche Produkt mit anderer, etablierter Technologie verdeutlicht diese Aussage: Es hätte zusätzlich zum Frontend ein Backend entwickelt werden müssen, welches die Geschäftslogik beschreibt und separat getestet werden müsste. Der Betrieb des Backends, beispielsweise durch Containerisierung hätte übernommen werden müssen, und im Erfolgsfall müsste die Skalierung durch Orchestrierungswerkzeuge wie Kubernetes implementiert werden. Auch der Betrieb der Datenbank, die Skalierung und weitere infrastrukturelle Themen wie Backups müssten selbst übernommen und automatisiert oder zumindest verantwortet werden. Auch die Entwicklung einer eigenen Designsprache hätte weitere Kapazitäten und Kompetenzen erfordert, was zu einer größeren Abhängigkeit von Anderen und einer verlängerten Entwicklungszeit geführt hätte. Insgesamt hätte die Entwicklung dann ein Team von Entwicklern, Desi-

gnern und Domänenexperten erfordert, was neben den gesteigerten Kosten auch die Komplexität effizienter Kommunikation und Kollaboration eingeführt hätte.¹

¹Sicherlich hätte die Entwicklung des Produkts als Team auch eine Reihe von Vorteilen, darauf liegt an dieser Stelle jedoch nicht der Fokus.

9

Ausblick

In diesem Kapitel geht es darum, den Blick auf die Potentiale und zukünftigen Schritte von Lightbulb Learning zu richten. Diese ergeben sich teilweise aus den Änderungen der Rahmenbedingungen, wie in Form von technologischen Weiterentwicklungen, und teilweise aus den fachlichen Erkenntnissen, welche sich erst im Laufe der Entwicklung ergeben haben.

9.1. Technologischer Ausblick

9.1.1. Supabase Edge Functions mit Deno Deploy

Einen Monat vor dem Abgabetermin der Thesis veröffentlichte Supabase die Ergänzung ihrer Dienstleistung um Edge Functions mit Deno Deploy ¹. Auch wenn keine umfängliche Evaluation dieser neuen Möglichkeit durchgeführt werden konnte, lassen sich doch einige Schlüsse daraus ziehen. So ist macht Supabase beispielsweise seinen Entscheidungsprozess für die serverseitige Codeausführung zwischen Container-as-a-Service und Function-as-a-Service transparent: „Our first critical decision was to choose Functions as a Service (”FaaS”) over Containers and other alternatives. While Containers can allow for more flexible workloads, authoring and maintaining Functions allows for a substantially simpler developer experience-something Supabase has always focused on” [Par22]. Was hier aus der Perspektive von Supabase als vereinfachte Entwicklererfahrung bezeichnet wird, kann aus der Perspektive eines Entwicklungsteams als konkrete

¹<https://supabase.com/blog/2022/03/31/supabase-edge-functions>

Maßnahme zur Anwendung des 10. agilen Prinzips interpretiert werden: Der Maximierung des Aufwands den man nicht treibt. Mit der Veröffentlichung des Blog-Artikels, der diese neue Funktionalität beschreibt, nennt Supabase eine weitere grundlegende Entscheidung, nämlich die Unterstützung von Edge Functions: „[...] for Supabase [Edge] Functions, we decided to deploy far-and-wide so that they are as close to your end-users as possible“. Damit ist gemeint, dass die mit Supabase bereitgestellten Funktionen nicht in einem einzigen oder wenigen Rechenzentren laufen, etwa in der Nähe der Datenbank selbst, sondern stattdessen auf über 30 weltweit verteilten Rechenzentren. Dadurch sollen verkürzte Aufrufzeiten erreicht werden. Um Funktionen auszuführen, welche auf Datenbankaufrufe optimiert sind, könnten anstelle der Edge Functions die regulären Supabase Functions, wie sie in Kapitel 6.3.5 beschrieben werden, verwendet werden.

9.2. Fachlicher Ausblick

9.2.1. Der Kontext der Nutzung

Eine Frage, die während der Entwicklung und Erprobung von Lightbulb Learning immer deutlicher und wichtiger wurde, war die Frage nach dem Kontext, in dem Nutzer Lightbulb Learning verwenden. Damit ist beispielsweise gemeint, worauf Studenten reagieren, wenn sie sich einloggen und eine Frage formulieren. Bisher erschien dies der bewussten Erledigung einer Aufgabe zu entsprechen, und nicht eine natürliche Reaktion auf etwas Neues, das man gerade gelernt hat. Somit eröffnet sich die Frage, wie man die Nutzer dazu bringen kann, sich die Nutzung von Lightbulb Learning anzugewöhnen. Dieser Frage widmet sich der nächste Abschnitt.

9.2.2. Das Hakenmodell

Das 2014 erschienene Buch „Hooked: Wie Sie Produkte erschaffen, die süchtig machen“ [Eya14] von Nir Eyal beschreibt ein Hakenmodell, welches die Psychologie gewohnheitsbildender Produkte zusammenfasst und zugänglich macht. Dieses Modell besteht im Wesentlichen aus 4 Phasen, welche ein Benutzer eines solchen Produkts immer wieder durchläuft. Diese Phasen lauten Auslöser, Aktion, (variable) Belohnung und Investition. Die Auslösephase beginnt immer mit einem äußeren Auslöser (z.B. einer Benachrichtigung oder E-Mail), welche über einen längeren Zeitraum zum inneren Auslöser (z.B. aus der Langeweile heraus) und schließlich zur Gewohnheit wird.

Die Aktionsphase stellt die Reaktion auf den Auslöser dar. Hier sollte aus Produktsicht darauf optimiert werden, dass die Aktion möglichst reibungslos verläuft, um die Wiederholung der Aktion wahrscheinlicher zu machen. Im Fall von Lightbulb Learning könnte man hier beispielsweise das Formulieren und Veröffentlichen einer Frage betrachten. Für die Phase der variablen Belohnung beschreibt der Autor drei unterschiedliche Arten der Belohnung: Belohnungen des Stammes, der Jagd und des Selbst. Diese steinzeitlichen Begriffe werden bewusst so verwendet, um den Tiefenpsychologischen Ansatz des Modells zu unterstreichen. So sind Belohnungen des Stammes insbesondere soziale Anerkennung, welche aus dem Aktivwerden erwächst. Im Kontext von Lightbulb Learning könnte das der Anzahl der Likes für eine Frage oder Antwort entsprechen. Mit Belohnungen der Jagd sind Ressourcen wie Geld oder Informationen gemeint, welche man für die durchgeführte Aktion erhält - bei Lightbulb Learning wäre hier der Lerneffekt das entsprechende Äquivalent. Belohnungen des Selbst beziehen sich auf das eigene Gefühl nach Durchführung der Aktion, wie dem Gefühl der Vervollständigung bei einem Puzzle. Verstärken lassen sich alle Formen der Belohnungen durch die Anwendung der Variabilität. Fällt die Belohnung nicht immer gleich aus, sondern tritt mit bestimmten Wahrscheinlichkeiten in unterschiedlicher Intensität auf, so wird das Verlangen nach der Belohnung mit der Neugier verstärkt. Die vierte und letzte Phase ist die Investition: Hier geht es darum, den Kunden zum Einsatz von Ressourcen wie Geld, Zeit oder Mühe zu bringen, um den Wert des Produkts selbst wiederum zu erhöhen. Für Lightbulb Learning wäre dies beispielsweise das Verfassen von Antworten, das die Bindung von Nutzern zu Lightbulb Learning erhöht und die Partizipation an diesem Kurs für andere Teilnehmer wertvoller macht. Eine grafische Darstellung des Modells wird in Abbildung 9.1 gezeigt.

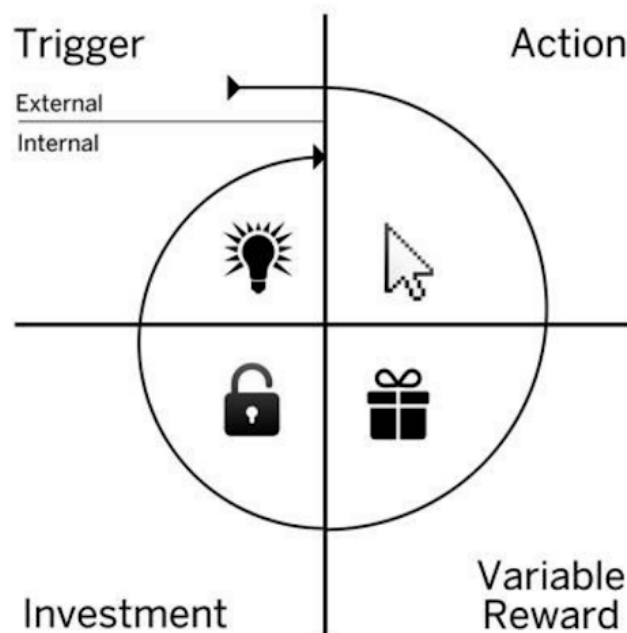


Abbildung 9.1: 4 Phasen des Hakenmodells: Auslöser, Aktion, variable Belohnung und Investition
[Eya14, S. 9]

Dieses Modell erweist sich für viele unterschiedliche Fallbeispiele als treffende Abbildung des menschlichen Verhaltens im Umgang mit sehr erfolgreichen digitalen Produkten, und könnte für Lightbulb Learning im Folgenden noch bewusster und unmittelbarer angewendet werden. Konkret bedeutet das beispielsweise, dass die äußeren Auslöser, die aktuell vollständig fehlen, in Form von automatisierten E-Mails umgesetzt werden könnten. Auch die variable Belohnung für die Interaktion mit dem System fehlt zum aktuellen Zeitpunkt noch. Hier könnte etwa mit animierten Fortschrittsbalken und Badges bei der Erreichung bestimmter Ziele Abhilfe geschaffen werden. Auch das Liken von Fragen und Antworten könnte mit einer schönen, grafischen Animation belohnt werden.

9.2.3. Legal-Themen

Ein bisher vollständig ausgeblendeter Bereich des Produkt Lightbulb Learning ist die Erfüllung rechtlicher Anforderungen wie einer Datenschutzerklärung und den Nutzungsbedingungen. Für dieses Aufgabe ist zum Zeitpunkt der Abgabe noch kein technologisches Werkzeug, etwa ein open source legal text generator oder ähnliches bekannt, so dass hier noch eine Strategie erdacht und umgesetzt werden muss.

9.2.4. Lightbulb Learning Zertifikate

Das Ziel für Lightbulb Learning, über diese Thesis hinaus, ist die wirtschaftliche Selbstständigkeit des Projekts. Die dafür benötigten Einnahmen sollen aus den Zertifikaten entstehen, wie es im Kapitel 4.2.5 bereits beschrieben wurde. Dieses Geschäftsmodell muss noch weiter validiert werden, indem mehr Feedback von Anbietern von Online-Kursen eingeholt wird, und auf dieser Grundlage das Feature mit Einbeziehung eines Bezahl-Dienstleisters umgesetzt wird. Ob Nutzer dafür bereit sind, sich auf Lightbulb Learning einzulassen, hängt im wesentlichen von zwei Faktoren ab: Glauben sie, dass sie mit Lightbulb Learning effektiv lernen können und glauben sie, dass das daraus resultierende Resultat von Wert ist. Mangelt es an der pädagogischen Überzeugtheit, so werden nur wenige Menschen überhaupt mit der Plattform interagieren. Fehlt es an subjektiver Wertschätzung der auszugebenden Zertifikate, so werden nach Erreichung der Qualifikation für das Zertifikat nur wenige Menschen dieses käuflich erwerben. Somit sind die Sicherstellung dieser zwei Aspekte die großen Herausforderungen für Lightbulb Learning nach der Thesis.

Literatur

- [Hew73] Carl Hewitt. "A universal modular ACTOR formalism for artificial intelligence". In: (1973).
- [Non97] Ikujiro/Hirotaka Takeuchi/Friedrich Mader Nonaka. *Die Organisation des Wissens: Wie japanische Unternehmen eine brachliegende Ressource nutzbar machen*. Campus Verlag, 1997.
- [Fie00] Roy Thomas Fielding. "Architectural Styles and the Design of Network-based Software Architectures". Diss. University of California, Irvine, 2000. URL: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm.
- [Bec01] Kent et al. Beck. *Prinzipien hinter dem Agilen Manifest*. 2001. URL: <https://agilemanifesto.org/iso/de/principles.html> (besucht am 30. 03. 2022).
- [Eva04] Eric Evans. *Domain-driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley, 2004.
- [Fow05] Martin Fowler. *Event Sourcing*. 2005. URL: <https://martinfowler.com/eaDev/EventSourcing.html> (besucht am 01. 04. 2022).
- [Nor06] Dan North. *Introducing BDD*. 2006. URL: <http://dannorth.net/introducing-bdd/> (besucht am 06. 04. 2022).
- [Pag07] Dave Page. *Project name - statement from the core team*. 2007. URL: <https://www.postgresql.org/message-id/473D7617.6070900@postgresql.org> (besucht am 29. 03. 2022).
- [Rot09] Mike Rother. *Toyota Kata: Managing People for Improvement, Adaptiveness and Superior Results*. McGraw-Hill Education, 2009. ISBN: 9780071639859.
- [Hew10] Carl Hewitt. *Actor Model of Computation: Scalable Robust Information Systems*. 2010. URL: <https://arxiv.org/pdf/1008.1459.pdf> (besucht am 11. 04. 2022).
- [Fow11] Martin Fowler. *CQRS*. 2011. URL: <https://www.martinfowler.com/bliki/CQRS.html> (besucht am 01. 04. 2022).

- [Rie11] Eric Ries. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown, 2011. ISBN: 9780307887917. URL: <https://books.google.de/books?id=tvfyz-4JILwC>.
- [Fit13] Rob Fitzpatrick. *The Mom Test: How to talk to customers & learn if your business is a good idea when everyone is lying to you*. A foundercentric.com book. CreateSpace Independent Publishing Platform, 2013. ISBN: 9781492180746.
- [Śle13] Karol Śledzik. "Schumpeter's View on Innovation and Entrepreneurship". In: (2013).
- [Bon14] Jonas/Dave Farley/Roland Kuhn/Martin Thompson Bonér. *Das Reaktive Manifest*. 2014. URL: <https://www.reactivemanifesto.org/de> (besucht am 08.02.2022).
- [Eya14] Nir Eyal. *Wie Sie Produkte erschaffen, die süchtig machen*. Redline Verlag, 2014.
- [Red17] Redhat. *Was ist Serverless?* 2017. URL: <https://www.redhat.com/de/topics/cloud-native-apps/what-is-serverless> (besucht am 02.02.2022).
- [Har18] Rich Harris. *Virtual DOM is pure overhead*. 2018. URL: <https://svelte.dev/blog/virtual-dom-is-pure-overhead> (besucht am 06.02.2022).
- [Meu18] Rob van der Meulen. *What Edge Computing Means For Infrastructure And Operations Leaders*. 2018. URL: <https://www.gartner.com/smarterwithgartner/what-edge-computing-means-for-infrastructure-and-operations-leaders> (besucht am 04.04.2022).
- [Har21] Rich Harris. *Jamstack Conf Session: Transitional Apps*. 2021. URL: <https://jamstackconf.com/talk/c17544108e71/transitional-apps/> (besucht am 07.02.2022).
- [Cou22a] Coursera Inc. *Take quizzes*. 2022. URL: https://www.coursera.support/s/article/209818703-Take-quizzes?language=en_US (besucht am 05.04.2022).
- [Cou22b] Coursera Inc. *Write Peer Reviews*. 2022. URL: https://www.coursera.support/s/article/209818803-Write-peer-reviews?language=en_US (besucht am 05.04.2022).
- [Gei22] Tim Geier. *Coursera Erfahrungen — Welche Zertifikate und Kosten?* 2022. URL: <https://www.e-learning-plattformen.de/coursera-erfahrungen-welche-zertifikate-gibt-es/> (besucht am 05.04.2022).

- [Par22] Inian Parameshwaran. *Edge Functions are now available in Supabase*. 2022. URL: <https://supabase.com/blog/2022/03/31/supabase-edge-functions> (besucht am 04.04.2022).
- [Pro22] Klaus D. Prof. em. Dr. Kubinger. *Stichwort: Multiple-Choice Antwortformat*. 2022. URL: <https://dorsch.hogrefe.com/stichwort/multiple-choice-antwortformat/> (besucht am 25.04.2022).
- [Sch22] Thor Schaeff. *Community Day*. 2022. URL: <https://supabase.com/blog/2022/03/28/community-day> (besucht am 04.04.2022).
- [Sup22a] Supabase. *Supabase: Functions*. 2022. URL: <https://supabase.com/docs/guides/database/functions> (besucht am 29.03.2022).
- [Sup22b] Supabase. *Supabase/README.de.md*. 2022. URL: <https://github.com/supabase/supabase/blob/master/i18n/README.de.md> (besucht am 16.02.2022).
- [Sve22] Svelte. *Svelte Tutorial, Reactivity / Declarations*. 2022. URL: <https://svelte.dev/tutorial/reactive-declarations> (besucht am 07.02.2022).
- [The22] The PostgreSQL Global Development Group. *PostgreSQL: About*. 2022. URL: <https://www.postgresql.org/about/> (besucht am 29.03.2022).
- [Ude22] Udemy, Inc. *Instructor Revenue Share*. 2022. URL: <https://support.udemy.com/hc/en-us/articles/229605008-Instructor-revenue-share> (besucht am 12.04.2022).
- [Wea22] Kelly Wearstler. *Transitional Design Guide: Characteristics of Transitional Design*. 2022. URL: <https://do6eyjibs3jse.cloudfront.net/articles/transitional-design-guide#4-characteristics-of-transitional-design> (besucht am 07.02.2022).

Abbildungsverzeichnis

Abbildung 2.1

Zieleinlauf des NYC Marathons 2018, Quelle: <https://www.nyrr.org/tcsnycmarathon/getinspired/photos-and-stories/marathon-goals-virtual-training>

Abbildung 2.2

Streckenprotokoll über Londoner Marathon, Quelle: <https://ultrastu.blogspot.com/2013/05/the-negative-split-fallacy-part-2.html>

Abbildung 3.1

Build - Measure - Learn Loop, Quelle: [Rie11, S. 81]

Abbildung 5.1

Die Darstellung der Abhängigkeiten der Qualitäten eines reaktiven Systems, Quelle: [Bon14]

Abbildung 6.1

High-level Architekturübersicht Lightbulb Learning

Abbildung 6.2

Row-Level-Security-Regel Beispiel für Kurs-Besitzer

Abbildung 6.3

Die join_course Funktion

Abbildung 6.4

Screenshot der Netzwerküberwachung in Chrome beim Löschen eines Kurses

Abbildung 7.2

Konzeptionelle Übersicht über einen Aufruf an ein CQRS System, Quelle: [Fow11]

Abbildung 7.1

Darstellung der Anzahl der Beiträge gruppiert nach Prüfling

Abbildung 9.1

4 Phasen des Hakenmodells: Auslöser, Aktion, variable Belohnung und Investition, Quelle: [Eya14, S. 9]

Danksagung

Ich danke allen Menschen, die mich bei der Erstellung dieser Arbeit, sowie des Produkts Lightbulb Learning, bis hierher unterstützt haben. Insbesondere bedanke ich mich bei Prof. Dr. Marko Boger und Prof. Jo Wickert für ihr Vertrauen und ihre Betreuung von Seiten der HTWG Konstanz. Vielen Dank an alle, die sich auf Experimente eingelassen und ihre Meinung gesagt haben, selbst dann, wenn die Wahrheit unangenehm war. Vielen Dank an alle, die sich für die Idee interessiert, sie mit mir diskutiert und ihre Ideen dazugegeben haben. Vielen Dank an meine Frau Katharina, die mich während der gesamten Zeit in jeder Hinsicht unterstützt, ermutigt und begleitet hat. Vielen Dank an Samuel Bach, für technologische Beratung und Unterstützung. Vielen Dank an Benjamin Vähjunker und Till Reitlinger für das Korrekturlesen von Teilen der Arbeit. Als initialen Funken für die Crowdsourcing-Idee für die Bewertung von Leistung möchte ich Dr. Jordan Peterson nennen, der in einer QA-Session am Ende eines Vortrags über Psychologie eine Antwort in diese Richtung gegeben hatte, und damit bei mir den Stein ins Rollen brachte.

Ehrenwörtliche Erklärung

Hiermit erkläre ich, Robert Nickel, geboren am 16.05.1993 in Warendorf,

1. dass ich meine Masterarbeit mit dem Titel:
„Skalierung einer Lern-Evaluationsplattform mit Serverless-Technologie“
in der Fakultät Informatik unter Anleitung von Professor Dr. Marko Boger selbständig
und ohne fremde Hilfe angefertigt habe und keine anderen als die angeführten
Hilfen benutzt habe;
2. dass ich die Übernahme wörtlicher Zitate, von Tabellen, Zeichnungen, Bildern und
Programmen aus der Literatur oder anderen Quellen (Internet) sowie die Verwen-
dung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb
der Arbeit gekennzeichnet habe.
3. dass die eingereichten Abgabe-Exemplare in Papierform und im PDF-Format
vollständig übereinstimmen.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Konstanz, 30.04.2022, _____