Leon Zimmermann

Github https://github.com/LeonZimmermann
LinkedIn https://www.linkedin.com/in/leon-zimmermann-5179831a4
Xing https://www.xing.com/profile/Leon_Zimmermann32/cv

Kurzübersicht

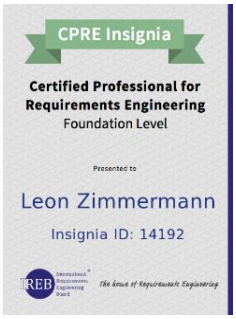
Ich habe in meiner Karriere bereits eine Vielzahl von Softwareprojekten umgesetzt. Mein bisher größter Meilenstein ist dabei die Entwicklung eines Abrechnungsportals für einen Kunden. Hier übernahm ich eine führende Rolle in der Entwicklung, konzipierte die Architektur der Software, stellte sicher, dass Qualitätskriterien eingehalten werden, koordinierte das Team und übernahm Deployments und die Kommunikation mit dem Kunden.

Schon mit 13 Jahren wusste ich, dass ich Softwareentwickler werden wollte. Ich habe mir damals autodidaktisch das Programmieren beigebracht und erste kleine Projekte umgesetzt. Ich habe mit einfachen CLI Programmen begonnen, und mir schon bald Java Swing angeeignet. Dann habe ich Spiele mit LWJGL entwickelt. Noch vor meinem Studienbeginn im Jahr 2018 habe ich erste Android Apps umgesetzt, darunter eine Vertretungsplan App für meine Schule. Während dem Studium habe ich 20 Stunden die Woche als Werkstudent gearbeitet, um zu lernen professionelle Software zu entwickeln. Die Verantwortung, welche ich in Projekten übernahm ist stetig gestiegen, womit ich heute an meinem aktuellsten Meilenstein angelangt bin.

Bildung und Beruf

Okt. 2018 - Okt. 2023	Universität Duisburg-Essen Studium Angewandte Informatik - Systems Engineering
März 2025 – Heute	Impressol GmbH
Sep. 2020 – Februar 2025	adesso SE
Sep. 2021 - Sep. 2022	tecis Finanzdienstleistungen AG
Juni 2019 - Okt. 2019	Maschinensucher.de

Zertifikate



Projektliste

(Januar 2025 – März 2025) Entwicklung einer Bestellschnittstelle (für Hörgeräte)

Der Produktdatenkatalog des Kunden wird umfassend erweitert, um als zentrale Bestellschnittstelle zu fungieren. Diese Erweiterung stellt sicher, dass die gesetzlichen Anforderungen an die elektronische Rechnungsstellung im B2B-Bereich vollständig erfüllt werden. Gleichzeitig zielt die Schnittstelle darauf ab, eine hohe Quote der Dunkelverarbeitung in der Bestellabwicklung zu ermöglichen. Durch die Automatisierung und Standardisierung der Prozesse wird eine effiziente und rechtssichere Abwicklung von Bestellungen gewährleistet, wodurch die Bearbeitungszeiten erheblich verkürzt und der Verwaltungsaufwand reduziert werden.

Technologien

- Docker
- Kubernetes
- Rancher for Desktop
- Taskfiles
- Helm Charts
- Minio
- Keycloak
- Hibernate
- JPA
- React
- TanStack Router

(September 2024 – Januar 2025) Entwicklung einer Plattform zur Verwaltung und Distribution von Produktdatenkatalogen (Hörgerätehersteller)

Um die Prozesse in der Hörakustik weiter zu verbessern, wird nun eine neue Plattform entwickelt, die es ermöglicht, die Produktdatenkataloge aller Hörgerätehersteller zentral zu verwalten. Diese Lösung wird eine einheitliche und standardisierte Struktur schaffen, sodass Hörakustiker jederzeit auf aktuelle und vollständige Informationen zu den Produkten zugreifen können.

Der Abruf dieser Kataloge erfolgt direkt die jeweiligen ERP-Systeme der Hörakustikerfachbetriebe.

Technologien

- Docker
- Kubernetes
- Rancher for Desktop
- Taskfiles
- Helm Charts
- Minio
- Keycloak
- Hibernate
- JPA
- React
- TanStack Router
- OAS Schema
- Testcontainers
- Postgres

- Spring Boot

Verantwortlichkeiten

- Aufsetzen von Minio im Cluster
- Implementierung von File System Storage
- Implementierung von API Keys
- Implementierung von Nutzerstatistiken
- Fullstack Implementierung von weiteren Features
- Implementierung von Testcontainers
- Erweiterung des OAS Schema

(2023 –2024) Entwicklung eines Abrechnungsportals

Das bestehende Abrechnungsportal des Kunden wird durch eine neue, zukunftsfähige Lösung ersetzt, welche für neue Prozesse im Geschäftsfeld des Kunden gerüstet und skalierungsfähig ist. Darunter zählt beispielsweise das Einbinden eines Klassifikationssystem für ein pauschaliertes Abrechnungsverfahren (DRG) sowie die Bereitstellung eines Datenaustauschverfahrens (DTA). Hierzu werden verschiedene Automatisierungs- und Digitalisierungspotenziale genutzt sowie eine leichtgewichtige Prozesssteuerung und Regelmaschine. Zusätzliche Features wie die Taskverwaltung unterstützen den Endnutzer und sein Team in der Aufgabenbewältigung. Dank einer responsiven grafischen Benutzeroberfläche (GUI) ist das neue Portal ebenfalls für Mobilgeräte geeignet.

Technologien

- Spring Boot
- Spring HATEOAS
- Postgres
- Testcontainers
- Angular
- Keycloak
- Hibernate
- SonarQube
- Nginx

Verantwortlichkeiten

- Erweiterung eines bestehenden Regelwerkes
- Konzeption und Implementierung eines neuen Regelwerkes
- Konzeption der Architektur und Implementierung von Softwarekomponenten
- Konzeption von Statemachines und Validierungsmechanismen
- Konzeption der Testing Architektur
- Konzeption, Aufsetzen und Durchführen von Lasttests zur Sicherstellung einer akzeptablen Performance des Systems
- Erweiterung des Imports und der Migration von ICD und OPS Codes um die Kreuz-Stern-Notation
- Erweiterung von Leistungskatalogen um ein Rabattsystem (Leistungskataloge bilden Selektivverträge mit Krankenkassen im System ab)
- Verbesserung der User Experience bei der Suche nach ICD und OPS Codes
- Anpassung der Schnittstelle zum Buchungssystem
- Implementierung neuer UI Komponenten
- Behebung von Bugs im Frontend und Backend
- Sicherstellung der Einhaltung von Qualitätskriterien mithilfe von Sonarqube und Lasttests
- Einbindung von Sonarqube in die Pipeline

- Einarbeitung von neuen Mitarbeitern
- Schreiben und Zuteilen von User Stories
- Durchführung von Deployments auf die Kundenumgebung

(2023) adesso: Entwicklung einer Suchfunktion für ICD- und OPS-Codes mit OpenSearch

Technologien

- Thymeleaf
- Spring Boot
- OpenSearch
- Docker
- Docker-Compose

Verantwortlichkeiten

- Konzipierung der Software
- Aufsetzen einer Umgebung mit docker compose
- Aufsetzen und Konfigurieren von OpenSearch
- Entwicklung einer REST-Schnittstelle in Spring Boot
- Anbindung der Spring Boot Anwendung zu OpenSearch

MDK

(2020 - 2023) adesso MDK-Projekt: Entwicklung einer Branchensoftware für den Medizinischen Dienst

Die Medizinischen Dienste (MD) haben die adesso SE mit der Entwicklung einer neuen Branchensoftware beauftragt, die die bestehenden Einzelsysteme der jeweiligen Dienste ablösen soll. Für die Initiierung und Steuerung des Projekts wurde eigens die IT-Einheit MD-IT GmbH gegründet. Das initiale Volumen des Großprojektes beläuft sich auf rund 21 Mio. Euro zzgl. CR-Volumen und beinhaltet auch einen Wartungsvertrag über acht Jahre. Die neue Software wird die 15 Medizinischen Dienste dabei unterstützen, ihren Beratungsauftrag für die gesetzliche Krankenversicherung in Deutschland zu erfüllen. Dazu zählen folgende Nutzungskontexte: medizinische und pflegefachliche Beratungs- und Gutachterdienste, wie z.B. die Prüfung von Krankenhausrechnungen, von medizinischen Verordnungen und Reha-Leistungen sowie die Begutachtung der Pflegebedürftigkeit mit Zuordnung von Pflegegraden. Dazu kommunizieren die Medizinischen Dienste in den Bundesländern mit ihren Auftraggebern, den gesetzlichen Krankenkassen, sowie mit Ärzten, Krankenhäusern und Versicherten.

Technologien:

- **Backend:** Spring Boot, REST API, Keycloak, Hyperdoc, Hazelcast
- **Frontend:** Angular, Bootstrap
- **Persistenz:** Postgres
- **Infrastruktur:** Docker, Docker-Compose, Ansible
- **Lasttests:** Gatling

Verantwortlichkeiten:

- Umsetzung eines Tools zur Diagnose eines Hazelcast-Clusters: Im Rahmen des Projekts wird ein Hazelcast Cluster zum Cachen von Daten und als gemeinsamer Speicher zwischen SCSen verwendet. Jede SCS hat einen eigenen Hazelcast Cluster, welcher Daten für diese spezifische SCS speichert, und zusätzlich gibt es einen globalen Hazelcast Cache, welcher Daten zwischenspeichert, die zwischen den SCSen geteilt werden. Das Diagnosetool sucht die vorhandenen Hazelcast Cluster und stellt eine Verbindung zu ihnen her. Anschließend fragt das Diagnosetool die Daten ab, die in den Clustern gespeichert sind. Über eine Oberfläche, die mit Thymeleaf und Bootstrap implementiert wurde, können die Daten eingesehen und manipuliert werden.

- Neue Controller in Spring implementiert und über das API-Gateway erreichbar gemacht, indem das forwarding in der config von nginx angepasst wurde. Außerdem die SecurityConfiguration in Spring angepasst und dort den Zugriff auf den Endpunkt erlaubt.
- Verwendung von Prometheus und Grafana für Monitoring
- Implementierung eines Kompatibilitätschecks zwischen Backend und Frontend
- Anpassungen von Ansible Playbooks
- Automatisierung von E2E-Tests
- Schreiben von Unit- und Integration-Tests
- Erstellung von Hyperdoc Templates mit XHTML, XSLT und DFF
- Gemeinsame Planung und Umsetzung von Lasttests mithilfe von Gatling
- Verwendung von XPath im Rahmen der Lasttests

(2023) Implementierung und Evaluierung einer Semantischen Suche

Im Rahmen meiner Arbeit habe ich Confluence als Wissensdatenbank verwendet. Confluence bietet eine Suchfunktion an, welche auf Apache Lucene basiert. Die Suchfunktion empfand ich persönlich als schlecht umgesetzt. Ich habe weitere Nutzer über dessen Meinung zu der Suchfunktion von Confluence befragt, und kam zu dem einheitlichen Ergebnis, dass die Suchfunktion häufig nicht die gesuchten Dokumente findet. Mit meiner Bachelorarbeit habe ich eine semantische Suche umgesetzt, um dieses Problem zu lösen. Anschließend habe ich auf Basis von echten Confluence-Daten einen Vergleich der Precision der Confluence-Suche mit der Precision der semantischen Suche durchgeführt. Das Ergebnis des Vergleichs war, dass die semantische Suche noch wesentlich schlechtere Ergebnisse liefert als die Confluence-Suche. Dies ist zurückzuführen auf die Tatsache, dass es sich bei den Daten um eine Closed-Domain handelt. Das Fine-Tuning von Transformern und Domain Adaption sind in der Literatur Ansätze, um dieses Problem zu lösen.

Technologien:

- **Backend:** Spring Boot, REST API, Gson
- **Persistenz:** Weaviate
- **Infrastruktur:** Docker, Docker-Compose
- **Natural Language Processing:** Sentence Transformer, OpenNLP

Verantwortlichkeiten:

- Implementierung einer REST API als Wrapper für Weaviate
- HTML-DOM-Parsing, Stopword-Removal, Stemming und Tokenization im Rahmen des Preprocessings der Confluence HTML-Dokumente
- Anbindung an Weaviate
- Konfigurierung der Vektordatenbank und der Sentence-Transformer
- Aufsetzen einer Test-Umgebung mit Docker-Compose und Gradle
- Implementierung von Unit- und Integration-Tests
- Implementierung eines Integration-Tests zur automatischen Durchführung der Evaluierung

(2023) Automatische Generierung von SQL-Übungsaufgaben

Im Rahmen meines Bachelorprojektes an der Uni habe ich eine Software entwickelt, welche automatisch SQL-Übungsaufgaben generiert. Diese Übungsaufgaben sollen es Studenten ermöglichen mit abwechslungsreichen Übungsaufgaben effektiv Konzepte aus der Query-Language zu lernen. Die Software besitzt zum Generieren keine Datengrundlage. Das Lehrpersonal gibt der Software über eine REST API eine URL zu einer Ontologie. Auf Grundlage der Ontologie wird ein Datenbankschema generiert, eine Datenbank aufgesetzt und die Datenbank mit automatisch generieren Testdaten befüllt. Der Student kann über die REST API eine Übungsaufgabe für diese Datenbank anfragen. Die Software

generiert daraufhin einen zufälligen Abstract Syntax Tree. Dieser Abstract Syntax Tree wird dazu verwendet, um die natürlichsprachliche Aufgabenstellung zu generieren, sowie eine SQL-Musterlösung. Die Lösung des Studenten wird mit der SQL-Musterlösung abgeglichen, indem beide Queries gegen die Datenbank ausgeführt werden. Liefern beide Queries das gleiche Ergebnis, dann ist die Query korrekt.

Technologien:

- **Backend:** Spring Boot, REST API
- **Persistenz:** Postgres
- **Infrastruktur:** Docker, Docker-Compose
- **Natural Language Processing:** Apache Jena, SimpleNLG, Ontologien
- **Sonstiges:** Abstract Syntax Trees

Verantwortlichkeiten:

- Konzeption der Software
- Parsing einer Ontologie in ein Datenbankschema mithilfe von Apache Jena
- Entwicklung eines Abstract Syntax Trees zur Generierung von SQL-Musterlösungen und die entsprechende Aufgabenstellung
- Generierung einer sprachlich korrekten Aufgabenstellung mithilfe von SimpleNLG
- Implementierung einer Clean Architecture nach Robert C. Martin

(2020) Wir-vs-Virus Plucky: Eine Plattform zur Vermittlung zwischen Landwirten und Kurzarbeitern

Während der Corona-Zeit haben viele Studenten ihre Nebeneinkünfte verloren. Sie haben beispielsweise als Kellner in der Gastronomie ihr Geld verdient, und konnten dies nicht weiter tun, als die Gastronomie geschlossen wurde. Gleichzeitig hatten Landwirte einen Mangel an Kurzarbeitern. Üblicherweise kommen Kurzarbeiter aus dem Ausland nach Deutschland, um hier gutes Geld zu verdienen. Durch Corona war dies nicht mehr möglich. Plucky war ein Prototyp für eine Vermittlungsplattform zwischen Landwirten und potenziellen Kurzarbeitern, u.a. Studenten.

Technologien:

- **Frontend:** Angular, Material Design
- **Backend:** Firebase

Verantwortlichkeiten:

- Implementierung von Services für die Anbindung an die Firebase-Datenbank
- Implementierung der Geschäftslogik für verschiedene Views
- Umsetzung von Designs für verschiedene Views

(Juni 2019 – Oktober 2019) Maschinensucher.de

Maschinensucher.de ist eine Website zum Inserieren von Gebrauchtmaschinen. Die Website kann als ein Google Kleinanzeigen für Maschinen betrachtet werden. Als Werkstudent habe ich die Website gewartet und weiterentwickelt.

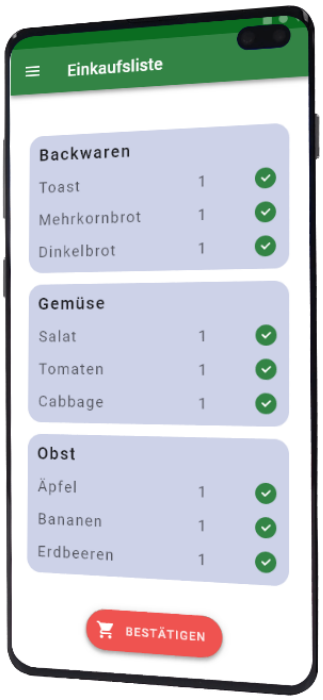
Technologien:

- **Frontend:** JavaScript/CSS/HTML, Bootstrap
- **Backend:** PHP, Yii2
- **Persistenz:** MySQL

Verantwortlichkeiten:

- Weiterentwicklung und Instandhaltung der Website Maschinensucher.de
- Migration von Teilen der Website von Yii1 auf Yii2

(2017) Usedup: Android App zur automatischen Erstellung von Einkaufslisten



Der Nutzer trägt seinen Bestand von Lebensmitteln und Kosmetikprodukten in der App ein, sowie den gewünschten Mindestbestand und eine Auffüllmenge der Produkte. Wenn die Menge eines Produktes unter dem Mindestbestand liegt, dann wird das Produkt zur Einkaufsliste hinzugefügt. Mit dem Einkauf wird der Bestand des Produktes wieder auf die Auffüllmenge aufgefüllt. Über die Oberfläche kann registriert werden, wenn ein Produkt aufgebraucht wird. Es können Gerichte in der App eingetragen werden. Mit dem Kochen des Gerichtes wird der Bestand der verwendeten Produkte automatisch aktualisiert.

Technologien:

- **Frontend:** Kotlin, Coroutines, Hilt – Dependency Injection Framework, Android
- **Backend:** Firebase
- **Infrastruktur:** Gradle (inkl. Multi-Project Builds), Git

Verantwortlichkeiten:

- Design der Benutzeroberfläche
- Design einer Marketing-Website für die App
- Aufsetzen eines Firebase-Projektes
- Konfiguration von Firebase Authentication und Firebase Firestore
- Implementierung eines Firebase API-Wrappers in Android, mithilfe von Kotlin Coroutines
- Verwendung des MVVM-Patterns
- Reaktive Programmierung mit LiveData
- Verwendung von Glide als Image Loading Framework
- Implementierung der Navigation zwischen Views
- Implementierung von Notifications
- Implementierung von Unit-Tests

(2016) Block Breaker: Ein Breakout-Klon im Rahmen des Informatik-Kurses meiner Schule (Java)

(2016) Entwicklung eines Android-Wrappers für den Online-Vertretungsplan meiner Schule

Das Theodor-Heuss-Gymnasium in Essen-Kettwig besaß zu meiner Schulzeit einen Online-Vertretungsplan. Ich war unzufrieden mit der Darstellung der Daten auf dem Smartphone, sodass ich einen Android-Wrapper geschrieben habe, welcher die gleichen Daten darstellte, aber auf eine für mich übersichtlichere Weise. Mit dem Projekt wollte ich erste praktische Erfahrungen mit Android sammeln.

Technologien:

- Java
- Android
- Gradle

Verantwortlichkeiten:

- Implementierung eines Webscrapers
- Implementierung des Login-Prozesses
- Design einer Benutzeroberfläche
- Mapping der Daten aus dem Webscraper auf die interne Datenstruktur
- Darstellung der gemappten Daten in einer Tabelle
- Implementierung einer Detailansicht

(2015) Starship: Ein Space Invaders ähnliches Spiel

Der Spieler steuert das blaue Raumschiff. Er kann nach links und rechts fliegen, um den Schüssen und Explosionen der gegnerischen Raumschiffe auszuweichen. Mit der Leertaste schießt der Spieler Laser ab. Durch das Sammeln von Credits kann sich der Spieler Upgrades im Shop kaufen. Ein stärkerer Laser, ein Schutzschild, Lebensregeneration. Ziel des Spiels ist es so lange zu überleben, wie möglich, und dabei einen möglichst hohen Highscore zu erzielen.

Technologien:

- Java
- LWJGL2
- OpenAL

Verantwortlichkeiten:

- Vollständige Konzeption des Spiels
- Design der Grafikelemente
- Erstellung von Soundeffekten
- Implementierung von Renderers mithilfe von Vertex- und Fragment-Shaders
- Implementierung einer selbst konzipierten KI zur Steuerung der Gegner im Spiel
- Implementierung der Benutzerschnittstelle
- Implementierung von visuellen Effekten (Explosionen, Parallax-Effekt)
- Persistenz von Highscores

(2013) CoinCollector: Ein Snake ähnliches Spiel

Der Spieler bewegt mit den WASD-Tasten ein Rechteck und muss Münzen einsammeln. Dabei darf der Spieler den Rand des Spielfelds nicht berühren. Die Schwierigkeit des Spiels besteht darin, dass das Rechteck sich ausgesprochen schnell bewegt und nicht gestoppt werden kann. Eine Berührung des Bildschirmrandes führt zu einem Game Over. Das Einsammeln von 20 Münzen führt zu einem Sieg des Spiels.

Technologien:

- Java

Verantwortlichkeiten:

- Vollständige Konzeption des Spiels
- Implementierung der Spiellogik