Sebastian Isaak

15. September 1991

Grenzweg 3c
32425 Minden

T +49 176 706 578 79
E sebastian.isaak@gmx.de
H **www.sebastian-isaak.de**

Profil

Als Softwareentwickler und -architekt arbeite ich seit einigen Jahren an Projekten für kleine bis große Unternehmen. Ich werde als sehr kommunikativ und anpassungsfähig beschrieben, weswegen mir die Onboardingphase im Unternehmen und im Team liegt. Durch die verschiedenen Projekte, die ich bislang begleiten durfte, konnte ich mir bereits früh ein breites Skillportfolio aneignen und dieses über die Jahre stetig weiter ausbauen.

Dabei steht für mich nicht nur Cleancode im Vordergrund, der einen verständlichen und gut strukturierten Code voraussetzt, sondern vor allem auch das KISS-Konzept. Der voranschreitende Druck der Digitalisierung und die dafür individuellen Lösungen der Unternehmen, setzen eine immer zunehmende Komplexität der Softwareentwicklung, der Softwarearchitektur und deren Infrastruktur voraus. Umso wichtiger ist es, technische Lösungen einfach und gut verständlich zu halten, um eine reibungslose Übergabe zu ermöglichen.

Technischer Fähigkeiten

Sprachen
Java
Typescript

Frameworks
Spring Boot
Java MicroProfile
JPA / Hibernate
Lombok
SLF4J

Schnittstellen
Rest
GraphQL
+ Apollo Client

Virtualisierung
Docker
Kubernetes

Datenbanken
SQL Server
MongoDB

Develop GUI
IntelliJ IDEA
Eclipse
Android Studio

Projekt Tools
Azure DevOps
Atlassian Stack
GitLab

Testing
JUnit / Jest
Cucumber
Selenium
Cypress

Vorgehensm.
Scrum

Frontend
React
JSF & PrimeFaces

Weitere Tools
Azure Cloud
Google Cloud Platform
Keycloak
IoTHub
Serverless Functions
Blobstorage
Terraform
Maven / Gradle
Grafana
SonarQube
Elastic
Kafka & RabbitMQ
Wildfly

Berufserfahrung

02/2023 - 08/2024 | **Software-Engineer,** REWE digital GmbH

Projekt „Authentication & Authorization Platform“:
Im Konzern der REWE Group war das Produktteam „Authentication & Authorization Plattform“ (kurz AAP) die zentrale Anlaufstelle für alle Authentifizierungsanfragen. Je nach Authentifizierungsart unterscheiden sich die Anbindungen und Authentifizierungsflows grundlegend (einige Beispiele: Benutzername und Passwort, NFC-Chips an Kassensystemen, Authenticator Apps, Passwordlose Authentifizierung, uvm.).
Um die Vielzahl der Anfragen (200 Millionen pro Woche) zu verarbeiten, sind mehrere Keycloak Instanzen in einem Kubernetes Cluster innerhalb der Google Cloud Platform bereitgestellt. Diese Keycloak Instanzen sind mit selbst entwickelten Erweiterungen bestückt, um die teils individuellen Authentifizierungsprozesse abbilden zu können. Angebunden sind die Keycloaks an mehreren Stammdatenverzeichnissen, um Benutzer verschiedener Tochtergesellschaften authentifizieren zu können.
Eine von vielen Erweiterungen kümmert sich um Audit Events, die neben in ein Kafka Topic geschrieben werden. Diese Events werden von diversen Backends konsumiert und Folgeaktionen durchgeführt.
Des Weiteren werden für den Keycloak diverse Spring Boot Backends im selben Cluster mit PostgreSQL Datenbanken bereitgestellt, die Entitäten für den Keycloak verarbeiten.
Da die zentrale Anwendung höchste Verfügbarkeit aufweisen muss, stehen neben üblichen JUnit und Integrationstests auch End-to-End Tests mittels Cypress zur Verfügung, die vor und nach Änderungen, Funktionen sicherstellen. Um zusätzliche Sicherheit und Verfügbarkeit zu gewährleisten, existieren weitere Keycloak Instanzen in Systemkritischen Einrichtungen wie zum Beispiel den Lägern der REWE. Hier wird mittels selbst implementiertem Backend per REST Ressourcen des zentralen Keycloaks in die dezentralen Keycloak dupliziert. Diese dezentralen Keycloaks stellen auch in einem Offlinefall die Möglichkeit sich in den Lägern anmelden zu können.
In einer Google Cloud Platform werden Cluster und Datenbanken gehostet, sowie Cloud Functions, die zu Logging Zwecken eingehende Anfragen protokollieren. Die Ressourcen innerhalb der GCP und im Keycloak werden mittels Terraform (Infrastructure as Code) verwaltet und erweitert.
Gearbeitet wird im agilen Kontext nach Scrum. Wobei DevOps Tätigkeiten in Kanban Boards festgehalten werden. Zum Team gehören 6 Backend Entwickler, 3 Frontend Entwickler, 1 Requirement Engineer, 1 Solution Architect, 1 Product Owner und 1 Scrum Master.

Angewendete Tools / Technologien: Java 17, Java Spring Boot 3.1, Keycloak, Google Cloud Platform, Kafka, Infinispan, JUnit, Rest, MySQL & PostgreSQL Datenbanken, Cypress, SonarQube, Docker, Kubernetes, Terraform, Maven, Confluence, Jira, GitLab.

12/2022 - 01/2023 | **Software-Engineer,** VBMC ValueBasedManagedCare GmbH

Projekt „Alley“:
In einem Greenfield wurde das Projekt Alley gestartet. Alley ist eine App, die Patienten bei Knie- und Hüftarthrose unterstützen soll. Patienten können von Ihrem Hausarzt einen Freischaltcode verschrieben bekommen, der den Zugang zu der App gewährleistet. In der App bekommt der Patient Tipps rund um die Behandlung, Präventionsmaßnahmen, wissenschaftlichen Artikeln und Fitnessübungen.
Da die App in der Medizinbranche eingesetzt wird, ist diese großen Qualitätsstandards unterlegt, die in regelmäßigen Abständen in Audits überprüft wurden. Eine genaue Entwicklerdokumentation, sowie diverse UML-Diagramme gehörten somit zur Bearbeitung von Tickets dazu.

Gearbeitet wurde agil nach Scrum. Zum Team gehörten vier Backend Entwickler, ein Scrum Master und ein Product Owner Team.
Das Frontend der App ist mit Angular entwickelt und vom Backend gekapselt. Das Backend ist microserviceorientiert und besteht aus rund acht Services. Jeder dieser Spring Boot (Version 2.7) Applikation ist in einer eigenen Dockerinstanz gebaut und in einem Kubernetes Cluster deployt. Die Kommunikation zwischen den Endpunkten wurde mit REST realisiert.
Es wurde mit der Versionsverwaltung GitLab gearbeitet. Für die Entwicklerdokumentation stand der Atlassian Stack zur Verfügung.
Um die hohen Qualitätsstandards der Medizinbranche und den Audits gerecht zu werden, wurden neben den Dokumentationen ein hoher Wert auf die Testabdeckung gelegt. Die Tests wurden größtenteils in JUnit und Cucumber geschrieben und deckten einzelne Methoden aber auch REST-Schnittstellen und Integrationstests ab. Diese wurden in der Pipeline mittels Sonarqube automatisch ausgewertet.
Als Identity Provider wurde Keycloak verwendet. Dieser wurde konfiguriert im Cluster deployt und diente zur Authentifizierung und Authorisierung aller Services.

Angewendete Tools / Technologien: Java 17, Java Spring Boot 2.7, Keycloak, JUnit, Rest, MySQL Datenbanken, Cucumber, SonarQube, Docker, Kubernetes, Terraform, Maven, Confluence, Jira, GitLab.

11/2021 - 11/2022 | **Software-Architekt,** symmedia GmbH

Projekt „SecureServiceHub“:
In einem Greenfield wurde das Projekt Secure Service Hub gestartet. Ziel dieser Webanwendung ist es, Produktionsmaschinen über eigeninstallierte Hardware zu vernetzen und die hochgeladenen Informationen Maschinenherstellern und/oder Kunden zur Verfügung zu stellen.
Der USP der Webanwendung war es über das Frontend 3rd Party Applications auf die EdgeDevices zu installieren, die an den Maschinen gekoppelt waren. Dabei wurden mittels Serverless Functions in der AzureCloud Manifeste und Webhooks an ein IoTHub übergeben und auf den gewünschten EdgeDevices die darin erhaltenen Docker Containern deployt. Der Status der Installationen wurde wiederrum über ein ChangeRequest vom IoTHub an eine weitere Serverless Function übergeben, in eine MongoDB geschrieben und im Frontend zur Laufzeit angezeigt.
Die Webanwendung selbst ist als Docker Container in einem Kubernetes Cluster deployt und wurde mit Typescript und React entwickelt. Im Backend stehen rund 25 weitere Microservices im Cluster bereit, die mit Java Spring Boot 2.7 entwickelt wurden. Für die Kommunikation zwischen den Services und dem Frontend wurde ein Gateway Service (Apollo Client) bereitgestellt, der die Föderation des GraphQL Schemas verwaltet.
Über den Standard OPC UA kommunizieren die Produktionsmaschinen mit Hilfe der vor Ort installierten Hardware. Auf der Hardware können über die entwickelten Webanwendungen Apps deployt werden, die in Docker Containern auf einem Linux System installiert werden.
Für die Qualitätssicherung wurden neben den JUnit / Jest Tests auch Integrationstest mittels Cucumber und E2E Tests mittels Cypress geschrieben.
Neben den genannten Services stehen auch viele weitere Ressourcen über die Azure Cloud zur Verfügung. Zum Beispiel: Azure SQL DB oder MongoDB für Backend Services, Kafka zur asynchronen Kommunikation, Blobstorages für die Dateiablage, das Azure Cluster mit allen notwendigen Keys und Routing Konfigurationen, Azure Functions als Serverless Service, IoTHub, Azure AD und AD B2C für die Benutzerverwaltung, Elastic für Monitoring und Logging, Azure Pipelines für ein automatisiertes Continuous Integration und Continuous Delivery, Azure DevOps für alle notwendigen Entwicklertools, App Registrations für die Verwaltung von außenstehenden Azure AD’s und viele weitere Ressourcen.

Um die Infrastruktur im Azure Portal sicherzustellen, wurden alle Ressourcen mit Terraform versioniert und angewandt. Dafür gab es für jede Umgebung der vier Systemlandschaften ein eigenes Directory im Azure Portal.
Gearbeitet wurde im agilen Umfeld mit 14-tägigen Sprints, einem Scrum Master, einem Product Owner und rund zehn Entwickler pro Team. Die Reviews fanden vor internationalen Stakeholdern auf Englisch statt.

Angewendete Tools / Technologien: Java 17, Java Spring Boot 2.7, Typescript, React, JUnit, Jest, GraphQL, Azure SQL Datenbanken, MongoDB, Cucumber, Cypress, Elastic, SonarQube, Kafka, Docker, Kubernetes (mit ArgoCD), Apollo Client, Terraform, Azure Cloud, Blobstorage, Maven, Kafka, IoTHub, Serverless Functions

10/2020 – 10/2021 | **Java Anwendungsentwickler,** Sänger.IO GmbH

Projekt „Topcube“:
Externe „Depots“ senden über eine Vielzahl von Schnittstellen Daten (zum Beispiel Ersatzteilbestellungen) im JSON-Format an das System. Die Schnittstellen für die Anwendung sind fachbezogen in Microprofiles entwickelt und laufen auf verschiedenen Serverinstanzen (Wildfly 16-22). Die angelegten Datensätze werden dann mithilfe eines „Taskservers“ anhand von mehreren Arbeitsschritten verarbeitet und in einer MySQL Datenbank gespeichert. Zur Verarbeitung zählen je nach Entität eine Vielzahl von Vorgängen (senden von Emails, Anlegen weiterer Daten, Benachrichtigung an andere Systeme über Schnittstellen, SMB-Shares lesen und schreiben, und vieles mehr). Zuletzt wurde das System um eine Schnittstelle erweitert, die sich um Abrechnungen kümmert. Dieses Projekt ist streng nach Microservice Architektur von der restlichen Software entkoppelt und speichert Daten in einem eigenen Datenbankschema.
Die Umsetzung erfolgte in einem Scrum Team bestehend aus fünf Entwicklern, einem Product Owner und einem Scrum Master.
Abwechselnd für jeden Sprint übernahm ein Entwickler die Rolle des Sprint-Springers. Dieser wurde nicht für die weitere Entwicklung im Sprint eingeplant, sondern kümmerte sich um das Operating. Somit konnte ich neben der Entwicklung viele Kenntnisse im Bereich DevOps, Bamboo Builds (CI / CD), Docker und automatisierte UI-Tests, WebService-Test und End-to-End Tests sammeln.

Angewendete Tools / Technologien: Java EE, Rest, JSF (PrimeFaces), MySQL, Atlassian Stack (Bitbucket, Jira, Bamboo, Confluence), Selenium, Wildfly, JPA / Hibernate, Java MicroProfile, Lombok, JUnit, SonarQube, Docker, Grafana, Maven

10/2018 – 09/2020 | **Softwareentwickler,** BEIT GmbH

Projekt „Web-to-go“:
Über ein selbstentwickeltes Frontend (JSF Primefaces) werden Tickets für Außendiensttechniker zur Verfügung gestellt. Diese Tickets werden in eine MobiLink Datenbank gespeichert und repliziert. Die Techniker sind in der Lage am Frontend Tickets zu bearbeiten, in Anfahrt zu nehmen oder auch neue Tickets zu erfassen. Über das System werden auch weitere Funktionen wie Arbeitszeit, derzeitiger Aufenthalt oder auch Reisekostenabrechnung zur Verfügung gestellt.

Projekt „Freischaltcode:
Über eine Weboberfläche (Liferay Portal) werden Anfragen an einen Server über selbstentwickelte Schnittstellen gesendet. Diese Anfragen werden in SQL-Datenbanken gespeichert und anhand von iDocs an SAP übermittelt. Die Schnittstellen sind

microserviceorientiert aufgebaut und die Entitäten entsprechen einer Domain Driven Design Architektur.

Projekt „Lagerverwaltung“:
In der Logistik werden Handscanner verwendet, auf denen ein Android Betriebssystem installiert ist. Diese Handscanner sind mit einer App ausgestattet, die es ermöglicht, Barcodes von Paletten und Materialien einzuscannen, weitere Informationen wie Position, Gewicht und Größe anzugeben und diese Informationen an SAP-Schnittstellen zu versenden.

Allgemeine Architektur der Projekte für die BEIT GmbH:
Die Anwendungen sind zum Großteil mit Java realisiert. In vielen Bereichen des Frontends wird aber auch Javascript verwendet.
Alle diese Projekte verfolgen einen objektorientierten Ansatz, besitzen getrennte front- und backend Systeme, eine Datenbank und oftmals Schnittstellen zu anderen Systemen wie zum Beispiel SAP.
Die Arbeitsweise ist an Scrum orientiert. Es wird in Sprints in JIRA gearbeitet, Confluence zur Dokumentation und GIT als Versionstool verwendet. Die Archive werden über einen Jenkins Server gebaut und auf Wildfly Servern publiziert.
Entwickelt werden diese Anwendungen mit Netbeans und Android Studio. Als Buildtool wird bevorzugt Maven verwendet. Im Android Studio wird auf das Buildtool Gradle ausgewichen.

Angewendete Tools / Technologien: Java EE, Javascript, JUnit, JSF / JSP, JPA / Hibernate, Jackson, SLF4J, AndroidStudio, Bitbucket, Confluence, Jenkins, Maven, Gradle, Wildfly

Berufsausbildung

Bachelor of Science, Studium Wirtschaftsinformatik	2016 – 2019
➔ Abschlussarbeit: asynchrone Kommunikation via RabbitMQ	
Fachinformatiker Systemintegration	2011 – 2014