IPA 2024 ABSCHLUSSARBEIT

***LEGACY DATEN IMPORT***

***von Raphael Blaauw***

Inhaltsverzeichnis

[Teil 1 - Umfeld und Aufbau 5](#_Toc158214468)

[1 IPA-Projektorganisation 5](#_Toc158214469)

[1.1 Projektorganisation 5](#_Toc158214470)

[2 Organisation der IPA-Ergebnisse 6](#_Toc158214471)

[2.1 Datensicherung 6](#_Toc158214472)

[2.1.1 Code 6](#_Toc158214473)

[2.1.2 Dokumentation 6](#_Toc158214474)

[3 Standards 6](#_Toc158214475)

[3.1 Technologien und Tools 6](#_Toc158214476)

[4 Zeitplan 9](#_Toc158214477)

[5 Arbeitsjournal 10](#_Toc158214478)

[5.1 Tag 1: Montag, 05.02.2024 10](#_Toc158214479)

[5.2 Tag 2: Dienstag, 06.02.2024 11](#_Toc158214480)

[5.3 Tag 3: Mittwoch, 07.02.2024 13](#_Toc158214481)

[5.4 Tag 4: Donnerstag, 08.02.2024 14](#_Toc158214482)

[5.5 Tag 5: Freitag, 09.02.2024 15](#_Toc158214483)

[5.6 Tag 6: Montag, 12.02.2024 16](#_Toc158214484)

[5.7 Tag 7: Dienstag, 13.02.2024 16](#_Toc158214485)

[5.8 Tag 8: Mittwoch, 14.02.2024 17](#_Toc158214486)

[5.9 Tag 9: Donnerstag, 15.02.2024 18](#_Toc158214487)

[5.10 Tag 10: Freitag, 16.02.2024 19](#_Toc158214488)

[6 Projektvorgehen 20](#_Toc158214489)

[6.1 IPERKA Methode 20](#_Toc158214490)

[6.2 Kanban 20](#_Toc158214491)

[Teil 2 – Projekt 21](#_Toc158214492)

[Kurzfassung 21](#_Toc158214493)

[Ausgangssituation 21](#_Toc158214494)

[Umsetzung 21](#_Toc158214495)

[Ergebnis 21](#_Toc158214496)

[7 Informieren 22](#_Toc158214497)

[7.1 Ziele der Aufgabenstellung 22](#_Toc158214498)

[7.2 Vorgaben 22](#_Toc158214499)

[7.3 Ist-Zustand 22](#_Toc158214500)

[7.3.1 Überblick 22](#_Toc158214501)

[7.3.2 Frontend 23](#_Toc158214502)

[7.3.3 Backend 23](#_Toc158214503)

[7.3.4 Klassendiagramm 24](#_Toc158214504)

[7.3.5 Datenbankdiagramm 25](#_Toc158214505)

[7.3.6 Systemgrenzen 25](#_Toc158214506)

[7.4 Analysieren der Probe-Excels 25](#_Toc158214507)

[7.5 Aktivitätsdiagramm 26](#_Toc158214508)

[7.6 User Stories 26](#_Toc158214509)

[7.6.1 Rollen 27](#_Toc158214510)

[7.6.2 Funktionale Anforderungen 27](#_Toc158214511)

[7.6.3 Nichtfunktionale Anforderungen 27](#_Toc158214512)

[8 Planen 29](#_Toc158214513)

[8.1 Soll-Zustand 29](#_Toc158214514)

[8.1.1 Überblick 29](#_Toc158214515)

[8.1.2 Frontend 29](#_Toc158214516)

[8.1.3 Backend 30](#_Toc158214517)

[8.1.4 Klassendiagramm 30](#_Toc158214518)

[8.1.5 Datenbankanpassungen 31](#_Toc158214519)

[8.2 Mockups 31](#_Toc158214520)

[8.3 Testkonzept 33](#_Toc158214521)

[8.3.1 Testziele und -strategie 33](#_Toc158214522)

[8.3.2 Testumgebung 33](#_Toc158214523)

[8.3.3 Testfälle 33](#_Toc158214524)

[9 Entscheiden 37](#_Toc158214525)

[9.1 Mockups 37](#_Toc158214526)

[9.2 Lösungsvarianten im Vergleich 37](#_Toc158214527)

[10 Realisierung 37](#_Toc158214528)

[10.1 xx 37](#_Toc158214529)

[10.2 yy 37](#_Toc158214530)

[10.3 zz 37](#_Toc158214531)

[Code Beispiel: 37](#_Toc158214532)

[10.4 Datenbank blah blah 37](#_Toc158214533)

[11 Kontrollieren 39](#_Toc158214534)

[11.1 Testing 39](#_Toc158214535)

[11.1.1 Testkonzept 39](#_Toc158214536)

[11.1.2 Testprotokoll 39](#_Toc158214537)

[11.1.3 Testbericht 39](#_Toc158214538)

[11.2 Unittests 39](#_Toc158214539)

[12 Auswerten 40](#_Toc158214540)

[12.1 Vergleich Ist / Soll 40](#_Toc158214541)

[12.2 Persönliches Fazit zur IPA 40](#_Toc158214542)

[12.2.1 Was war gut? 40](#_Toc158214543)

[12.2.2 Was lief weniger gut? 40](#_Toc158214544)

[12.3 Schlussreflexion 40](#_Toc158214545)

[13 Glossar 41](#_Toc158214546)

[14 Quellenverzeichnis 41](#_Toc158214547)

[14.1 Einsatz von generativen Tools 42](#_Toc158214548)

[15 Anhang 42](#_Toc158214549)

[15.1 Sitzungsprotokolle 42](#_Toc158214550)

[15.1.1 Erster Expertenbesuch 42](#_Toc158214551)

[15.1.2 Zweiter Expertenbesuch 42](#_Toc158214552)

# Teil 1 - Umfeld und Aufbau

# IPA-Projektorganisation

## Projektorganisation

**Lehrbetrieb und Durchführungsort:**

Generali Personenversicherungen AG

Soodmattenstrasse 10

8134 Adliswil   
058 472 44 44

**Kandidat:**

Raphael Blaauw

Marchwartstrasse 39, 8038 Zürich

079 622 61 76  
raphael.blaauw@generali.com

raphaelblaauw24@gmail.com

**Auftraggeber:**

Daniel Sager

Generali Personenversicherungen AG

Soodmattenstrasse 10, 8134 Adliswil

daniel.sager@generali.com

**Berufsbildner/ Lehrfirma:**

Roger Sorg

Generali Personenversicherungen AG

Soodmattenstrasse 10, 8134 Adliswil

079 779 53 69  
roger.sorg@generali.ch

**Verantwortliche Fachkraft:** Roman Babenko

Generali Personenversicherungen AG

Soodmattenstrasse 10, 8134 Adliswil

roman.babenko@generali.com

**Hauptexperte:**

Yves Kaufmann

y.kaufmann@yagan.ch

078 872 38 84

**Nebenexperte:**

Christian Soldan

christian.soldan@ubs.com

078 901 73 83 (P)

# Organisation der IPA-Ergebnisse

## Datensicherung

Die Datensicherung ist wichtiger Bestandteil des Gelingens der IPA. Für die Sicherstellung der Arbeit wurde auf diverse Versionierung und Speicherungssysteme gesetzt.

### Code

Der Codestack ist auf Github ersichtlich und für die IPA wurde ein neuen Branch erstellt.

**Branch**: [Link](https://github.com/IT-Apprentices/Offerten-ReferenzApplikation-Backend/tree/Probe-IPA-Vervollst%C3%A4ndigung-Other)

Folgend der Verlauf des Repositories:

|  |
| --- |
|  |

### Dokumentation

Durch Microsoft Office wird der neuste Stand bei OneDrive automatisch gespeichert. Für zusätzliche Sicherheit wird ein Backup am Schluss jedes Tages auf einem seperaten Github Repository erstellt:

https://github.com/blauWhale/IPA

Folgend der Verlauf des Repository bzw. der Sicherung

|  |
| --- |
|  |

# Standards

## Technologien und Tools

Im Team werden mit folgenden Tools und Technologien gearbeitet:

|  |  |
| --- | --- |
| **Frontend** | |
| **Name** | **Einsatz** |
| Next.js | Frontend-Technologie |
| shadcn | Komponenten Bibliothek |
| Typescript | Frontendsprache |
| Prettier | Code Formatierung |
| React Testing Library | Tests |
| Storybook | Frontend Workshop |

|  |  |
| --- | --- |
| **Backend** | |
| **Name** | **Einsatz** |
| Springboot | Backend-Technologie |
| Java | Backendsprache |
| shadcn | Komponenten Bibliothek |
| Typescript | Frontendsprache |
| Google Java Styleguide | Code Formatierung |
| JUnit | Tests |
| Spring Boot Testing | Tests |

|  |  |
| --- | --- |
| **Daten Layer** | |
| **Name** | **Einsatz** |
| PostgreSQL | Datenbank-Technologie |
| Flyway | Datenbankversionierungstool |

|  |  |
| --- | --- |
| **Identity Provider** | |
| **Name** | **Einsatz** |
| OpenID Connect | Identitätsprotokoll |
| Keycloak | Identitätsmanagement |

|  |  |
| --- | --- |
| **Lokale Entwicklung** | |
| **Name** | **Einsatz** |
| Jetbrains IntelliJ | Entwicklungsumgebung |
| Docker | Containerisierungstool |

|  |  |
| --- | --- |
| **UI/UX Design** | |
| **Name** | **Einsatz** |
| Figma | Diagramme/Skizzen |
| Generali Styleguide | Styleguide |

|  |  |
| --- | --- |
| **Dokumentation und Admin** | |
| **Name** | **Einsatz** |
| Microsoft Office | Dokumentation |
| Google Chat | Kommunikation |
| ChatGPT | Textgenerierung, Informationen |
| Draw.io / Plantuml | UML-Digramme |

# Zeitplan

# Arbeitsjournal

## Tag 1: Montag, 05.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** | Aufbau IPA-Bericht  Zeitplan erstellen  Analysieren von Beispielsdaten  Dokumentieren des IST-Zustandes |
| **Erreichte Ziele** | Gute Grundlage im Bericht aufgebaut und IST-Zustand abgebildet |
| **Probleme** | Das Erstellen von UML-Diagrammen mit IntelliJ und PlantUML brauchte seine Zeit.  Der Zeitplan erforderte viel vorausdenken und Striktes abschätzen und einplanen von Aufgaben |
| **Hilfestellung** | - |
| **Ausserplanmässige Arbeiten** | - |
| **Reflexion** | *Was lief heute Positiv?*  Voller Vorfreude auf diese IPA konnte ich meine Aufgabe fassen und die Arbeit daran beginnen. Ich bin mit viel Energie in diese zehn Tage gestartet. Der Aufbau des IPA-Berichts ging sehr gut dank einer guten Vorarbeit/Vorlage. Das Arbeiten mit IPERKA bringt viele Vorteile mit sich. Mit dieser Methode wird ein klarer Weg eingeschlagen, der keine Abkürzungen oder Überspringen von Schritten wie ein klares Konzept oder Planung zulässt. Mir persönlich fehlt aber einen gesamten Überblick und Fortschritt des Projektes. Daher habe ich für die IPA-Zeit ein Kanban Board auf Notion.so eingerichtet mit den heruntergebrochenen Aufgaben des Zeitplans. So weiss ich ganz genau an was noch offen ist, an was ich gerade bin und was ich bereits erledigt haben.  *Was sehe ich kritisch?*  Der Zeitplan war herausfordernd und brauchte viel genaues Arbeiten, um die Aufgaben über die zwei Wochen richtig zu verteilen. Ich kam aber gut voran und konnte schnell als gedacht den Plan fertigstellen. Ich möchte morgen den Zeitplan meinem Experten, Herr Kaufmann, präsentieren, um Feedback einzuholen. Jeder Experte legt den Schwerpunkt bei der Planung anders.  *Meine Erkenntnisse von heute?*  Ich wollte so viel wie möglich heute angehen. Mir bleibt aber noch genügen Zeit und ich muss mich selbst nicht unnötig stressen. |
| **Nächste Schritte** | Einarbeiten in den Umgang mit Excels in Spring/Java  User-Stories und Akzeptanzkriterien  Aktivitätsdiagramm des Import-Prozesses |

## Tag 2: Dienstag, 06.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** | Arbeiten mit Excel-Sheets in Java  User-Stories und Akzeptanzkriterien schreiben  1.Expertenbesuch  Aktivitätsdiagramm |
| **Erreichte Ziele** | Die letzten Arbeiten in der Informationsphase sind abgeschlossen. Morgen kann ich zur Planung übergehen. |
| **Probleme** | Es war schwierig abzuschätzen, was eine User-Story ist und was eine einfache Anforderung bzw. Kriterium der IPA. Nicht alles gehört als User-Story erfasst. |
| **Hilfestellung** | Kurze Diskussion über den Zeitplan und Kriterien mit Roman. Er gab mir Feedback wie ich meinen Dokumentationsaufwand am besten gegenüber dem Experten aufzeigen kann. |
| **Ausserplanmässige Arbeiten** | Kleinere Anpassungen am Zeitplan nach dem ersten Expertenbesuch. |
| **Reflexion** | *Was lief heute Positiv?*  Ich konnte heute viele Arbeiten abschliessen. Ich musste an diversen Ecken in der Dokumentation ergänzen, schreiben oder Grafiken erstellen. Trotzdem verlor ich den Überblick nicht. Die ersten Versuche ein Excel mit Java bzw. der Apache POI Library zu arbeiten, gelangen sehr gut. Ich musste mich selbst bremsen, um nicht gleich loslege mit entwickeln. Der erste Expertenbesuch war sehr angenehm und ich bin froh einen erfahrenen und fairen Experten erwischt zu haben.  *Was sehe ich kritisch?*  Die User-Stories auszuarbeiten kostete viel Zeit und ich war mir bei manchen Stellen unsicher, ob eine Anforderung in der Aufgabenstellung auch als User-Story anzusehen ist. Ein gutes Beispiel sind die Kriterien bezüglich dem Testing. Ich verfasste bewusst keine Stories oder Akzeptanzkriterien für jede Testart die gefragt ist. Sonst würde es beim Testkonzept dazu kommen das getestet wird ob so und so getestet wird. Dies machte für mich für den Umfang der IPA weniger Sinn. Wiederum Themen wie Tracing und strukturiertes Logging als Story zu erfassen, macht für mich durchaus Sinn. Aber nur im Umfang dieser Arbeit. So bin ich sicher, dass ich die Kriterien am Schluss erfüllt habe.  *Meine Erkenntnisse von heute?*  Durch das Feedback des Experten merkte ich das mein Zeitplan nicht klar meinen Dokumentationsaufwand wiederspeigelt. Ich muss besser aufzeigen das ich genügend Zeit für meinen IPA-Bericht aufwende.  Nach den ersten Arbeiten fühle ich mich gut informiert und habe eine gute Grundlage, um jetzt die Planung anzugehen. Ich freue mich das ich einen guten Einstieg in die Arbeit gefunden habe und dass ich mich soweit konzentriert und zielgerichtet arbeiten kann. |
| **Nächste Schritte** | Festlegen des Soll-Zustands im Backend und Frontend  Mockup des Import Wizards in Figma |

## Tag 3: Mittwoch, 07.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** | Planen des Soll-Zustandes im Front- und Backend  Mockups des Import Prozesses in Figma angefangen zu gestalten. |
| **Erreichte Ziele** | Soll-Zustand des Projektes festgehalten. |
| **Probleme** | - |
| **Hilfestellung** | - |
| **Ausserplanmässige Arbeiten** | - |
| **Reflexion** | *Was lief heute Positiv?*  Das Entwickeln der Idee zum Soll-Zustand konnte ich dank einer guten Vorbereitung in der Informationsphase gelassen angehen. Ich entschied mich mit Bulletpoints zu Arbeiten und kam so gut voran. Die Mockups in Figma zu gestalten, machte mir Spass und ich bin bereits sehr zufrieden mit den ersten Ideen. Morgen möchte ich noch eine kleine Variation einbauen, um verschiedene Optionen anbieten zu können. Während dem Gestalten geht die Zeit wie im Flug und ich muss aufpassen das ich das Dokumentieren nicht vernachlässige.  *Was sehe ich kritisch?*  Das Exportieren von diversen Diagrammen für die Dokumentation von IST- bzw. SOLL-Zustand gelang nicht auf Anhieb. Immer wieder entdeckte ich einen kleinen Fehler oder wollte noch eine kleine Änderung vornehmen.  Morgen Nachmittag werde ich am Nachmittag mit dem Entwickeln beginnen. Ein Kollege teilte mir mit das der Pull-Request mit dem Modul zum Darstellen der Kunden im OVWEB immer noch nicht im Main-Branch hinein gemerged wurde. Ich bin auf Roman zu und teilte ihm mit das dies unbedingt Priorität finden muss. Ich mache mir keine Sorgen bin aber ein Wenig frustriert das der Main-Branch am dritten Tag meiner IPA noch nicht bereit für meine Entwicklungsarbeiten ist.  *Meine Erkenntnisse von heute?*  Mit den weiteren Erkenntnissen in der Planung kann ich mir nun immer mehr ein Bild machen, wie die Realisierung ausschauen wird. Die begonnen Mockups in Figma geben mir sogar bereits ein visuelles Bild wie ich die Implementierung des Import-Wizards gestalten möchten. |
| **Nächste Schritte** | Variation des Mockups anfertigen  Testkonzept erstellen  Entscheid für eine Variation und Begründen im IPA-Bericht  Start der Entwicklung |

## Tag 4: Donnerstag, 08.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** | Mockups in Figma abschliessen  Testkonzept anfertigen  Beginnen mit den Entwicklungsarbeiten |
| **Erreichte Ziele** | Planung und Entscheidungsphase ist so weit abgeschlossen  Die Entwicklungsarbeiten sind gestartet |
| **Probleme** | Heute Nachmittag ist mein Laptop abgestürzt. Die Fehlerquelle ist WSL 2. Ich ging mit meinem Problem kurz zu Roman. |
| **Hilfestellung** | Roman hat mir bei meinem plötzlichen Blackscreen geholfen in dem er die Ursache nachging. Wir bemerkten das WSL 2 nicht auf der neusten Version ist. Ich werde am ende des Tages ein Update durchführen. |
| **Ausserplanmässige Arbeiten** | Kurze Session zu den WSL 2 Problemen |
| **Reflexion** | *Was lief heute Positiv?*  Variation, Mockups machen Spass, wenig Zeit verschwenden für Entscheidung. Guter einstieg in das programmierne. Excel wird gelesen  *Was sehe ich kritisch?*  Frustrirend die probleme mit WSL. Müde konzentartion, musik hilft.  *Meine Erkenntnisse von heute?*  So wird’s aussehen, testen wird aufwendig, aber genug Zeit geplant. Allgemein gut im Zeitplan. Neue Erkentnisse bei den Testdaten |
| **Nächste Schritte** | Backend Validierung  Frontend Erklärseite und Upload Modul |

## Tag 5: Freitag, 09.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** |  |
| **Erreichte Ziele** |  |
| **Probleme** |  |
| **Hilfestellung** |  |
| **Ausserplanmässige Arbeiten** |  |
| **Reflexion** | *Was lief heute Positiv?*  *Was sehe ich kritisch?*  *Meine Erkenntnisse von heute?* |
| **Nächste Schritte** |  |

## Tag 6: Montag, 12.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** |  |
| **Erreichte Ziele** |  |
| **Probleme** |  |
| **Hilfestellung** |  |
| **Ausserplanmässige Arbeiten** |  |
| **Reflexion** | *Was lief heute Positiv?*  *Was sehe ich kritisch?*  *Meine Erkenntnisse von heute?* |
| **Nächste Schritte** |  |

## Tag 7: Dienstag, 13.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** |  |
| **Erreichte Ziele** |  |
| **Probleme** |  |
| **Hilfestellung** |  |
| **Ausserplanmässige Arbeiten** |  |
| **Reflexion** | *Was lief heute Positiv?*  *Was sehe ich kritisch?*  *Meine Erkenntnisse von heute?* |
| **Nächste Schritte** |  |

## Tag 8: Mittwoch, 14.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** |  |
| **Erreichte Ziele** |  |
| **Probleme** |  |
| **Hilfestellung** |  |
| **Ausserplanmässige Arbeiten** |  |
| **Reflexion** | *Was lief heute Positiv?*  *Was sehe ich kritisch?*  *Meine Erkenntnisse von heute?* |
| **Nächste Schritte** |  |

## Tag 9: Donnerstag, 15.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** |  |
| **Erreichte Ziele** |  |
| **Probleme** |  |
| **Hilfestellung** |  |
| **Ausserplanmässige Arbeiten** |  |
| **Reflexion** | *Was lief heute Positiv?*  *Was sehe ich kritisch?*  *Meine Erkenntnisse von heute?* |
| **Nächste Schritte** |  |

## Tag 10: Freitag, 16.02.2024

|  |  |
| --- | --- |
| **Tätigkeiten** |  |
| **Erreichte Ziele** |  |
| **Probleme** |  |
| **Hilfestellung** |  |
| **Ausserplanmässige Arbeiten** |  |
| **Reflexion** | *Was lief heute Positiv?*  *Was sehe ich kritisch?*  *Meine Erkenntnisse von heute?* |
| **Nächste Schritte** |  |

# Projektvorgehen

## IPERKA Methode

Die IPERKA-Projektmethode besteht aus sechs Schritten: Informieren, Planen,

Entscheiden, Realisieren, Kontrollieren und Auswerten. Sie ist vielseitig einsetzbar,

etwa in Lehrbetrieben, Schulen und im Alltag. Die Methode fördert ein strukturiertes

Vorgehen und betont die Bedeutung von Planung und Informationsbeschaffung

sowie die Auswertung nach Abschluss des Projekts.

In diesem Kapitel beschäftige ich mich mit dem Aufbau des Projektes. Das Projekt ist

nach der IPERKA-Methode aufgebaut, damit stets eine gegliederte Struktur im

Projektablauf ersichtlich ist. Hierzu sind die 6 Phasen erforderlich.

**Informieren:**

Der Auftrag und dessen Aufgabenstellung werden so gut wie möglich geklärt, die

Informationen werden gewertet und sortiert und wesentliche Punkte so früh wie

möglich erkannt sowie allfällige Fragen geklärt. Dies ist wichtig, um den Zeitplan

fertigzustellen und zum Verständnis was überhaupt erwartet wird.

**Planen:**

Um einen guten und strukturierten Projektablauf zu garantieren, muss das Projekt

präzise geplant werden. Wenn möglich werden mehrere Lösungswege für

komplexere Abläufe geplant. Dabei wird ein Testkonzept, sowie ein

Realisierungskonzept erstellt.

**Entscheiden:**

Die unterschiedlichen Lösungswege werden miteinander verglichen und davon wird

dann der bestmögliche ausgewählt. Dabei muss man sich überlegen ob die Ideen

sinnvoll sind und ob man es dann auch wirklich so umsetzen kann.

**Realisieren:**

Erst wenn der Ablauf klar definiert ist und alle Entscheidungen getroffen sind, wird

mit der Realisierung begonnen. Die Arbeitsabläufe werden stetig protokolliert und

die Ist-Werte werden im Zeitplan eingetragen.

**Kontrollieren:**

Die Resultate müssen so getestet werden wie es im Testkonzept steht, aber erst

wenn die Entwicklung vollständig beendet ist. Fehlgeschlagene Test werden hierbei

direkt korrigiert und behoben oder als Fehler vermerkt und dokumentiert.

## Kanban

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte BeschreibungEin Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte BeschreibungNeben der IPERKA-Methode setzte ich zwei Kanban Board ein, um den Überblick zu behalten. Zu einem für den Kriterienkatalog, damit ich erledigte Punkte abhacken kann und mich auf offene konzentrieren kann. Das zweite für die tägliche Aufgaben während der IPA. Ich verwende die Board Funktion von Notion.so.

Kanban für die Schlusskontrolle

Kanban für tägliche Arbeit

# Teil 2 – Projekt

# Kurzfassung

## Ausgangssituation

## Umsetzung

// Auf eigenen Prozess eingehen bzw. Methodik einfliessen lassen

## Ergebnis

In diesem zweiten Teil des IPA-Berichts zeige ich die Umsetzung des Projektes mit dem IPERKA Modell auf. Die verschiedenen Phasen dienen als Orientierung für die Struktur des weiteren Berichtes.

# Informieren

## Ziele der Aufgabenstellung

Ziel ist es Legacy Daten in Form von Excels (.xls Dateien) mittels einem Import Wizard in das OVWEB übernehmen zu können.

## Vorgaben

Es sollten doppelte Einträge vermieden werden durch Überprüfung der existierenden Kundedaten anhand folgender Kriterien:

* Nachname
* Vorname
* Geburtsdatum

Ist der Kunde bereits erfasst, sollte sein Datensatz lediglich aktualisiert werden.

## Ist-Zustand

Die Offertenverwaltung Web, kurz OVWEB, wird dazu dienen die Cloudfähigkeit der

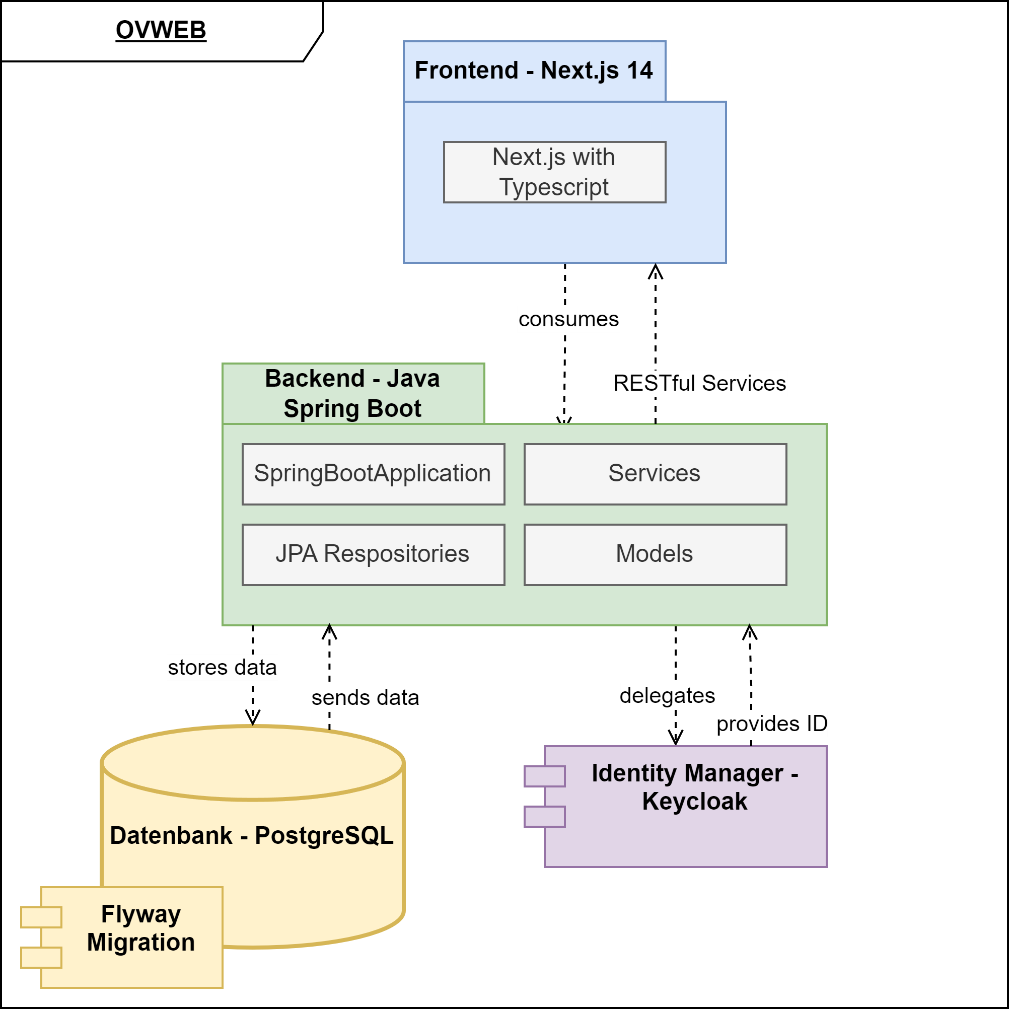
alten Applikation aufzuzeigen. Die Idee der Applikation ist Offerten zu erstellen und

Kunden zu verwalten. Der aktuelle Stand der Seite erlaubt das Login via Keycloak und

das Erstellen eines Kunden via ein Formular. Um nicht jeden Kunden einzeln im neuen Tool erfassen zu müssen, gibt es ein grosses Interesse bestehende Kunden via einen Export im alten Tool in das neue Tool importieren zu können. An dieser Anforderung der Berater setzte ich an, um den Transfer so einfach wie möglich zu gestalten. Ansetze zur Kunden Verwaltung gibt es ebenfalls.

### Überblick

Die folgende Abbildung gibt einen konzeptionellen Überblick über die Kernarchitektur der OVWEB-Anwendung. Sie konzentriert sich auf die wesentlichen Bestandteile.



Gesamtsystem OVWEB

### Frontend

Das Frontend wurde mit Next.js 14 umgesetzt und nach dem Generali Styleguide

gestaltet. Das Sessionhandling wird mit NextAuth geregelt. Als Component Library

wird ShadCn eingesetzt. Es wird ein TypeScript-Client für das Backend verwendet.

### Backend

Das Backend wurde mit Springboot aufgebaut. Ein simples Model wurde integriert das

User, Customer und ihre Addressen beinhaltelt. Flyway versioniert und initialisiert die

Datenbank Anbindung die lokal via einem Docker Container läuft. Als Query Sprache wird PostgreSQL verwendet.

### Ein Bild, das Text, Screenshot, 3D-Modellierung, Grafiksoftware enthält. Automatisch generierte BeschreibungKlassendiagramm

Model Klassendiagramm im IST-Zustand

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte BeschreibungFür das Customer und User Mode gibt es Service und Controller für CRUD-Operationen

Service Klassendiagramm im IST-Zustand

### Datenbankdiagramm

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte BeschreibungDie Datenbank wird durch Flyway verwaltet und enthält die gleichen Entitäten wie das Model im Backend

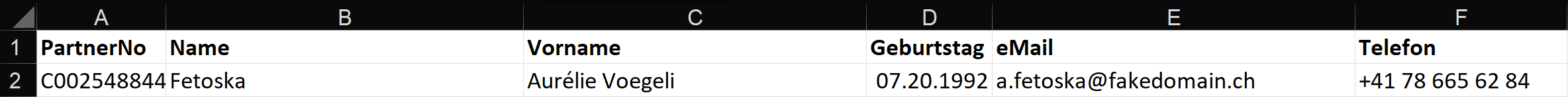
Datenbankdiagramm mit Flyway History

### Systemgrenzen

Der Auftrag kann in der vorhandenen Architektur der «OVWEB» eingebunden werden und braucht lediglich den Export der Kundendaten als Excel-Datei vorliegend. Für den Umfang der IPA werden diese Daten bereitgestellt und daher erfordert es keine Interaktion mit dem Legacy-Applikation.

## Analysieren der Probe-Excels

Beim Sichten der zur Verfügung gestellten drei Exceldaten «**AngebotSoftware\_Export\_TC001.xls**» (gute Daten), «**AngebotSoftware\_Export\_TC002.xls**» (gute Daten + doppelte Einträge), «**AngebotSoftware\_Export\_TC003.xls**» (Kopie von TC001 mit inkonsistenten Daten) sind mir folgende Punkte aufgefallen:

* Es wird das alte Dateiformat «.xls» verwendet
* Die Spalte «PartnerNo» ist nicht in unserem Model vorhanden und muss ergänzt werden.
* Die Formatierung des Geburtsdatums muss eventuell konvertiert werden
* Bei den nicht konsistenten Daten gibt es im «Adresses»-Sheet falsche «PartnerNo»

Beispielzeile des «Customers»-Sheet

Beispielzeile des «Adresses»-Sheet

Diese Erkenntnisse helfen mir eine saubere Planung des Import-Wizards für diese und ähnlich Datensätze zu gestalten.

## Aktivitätsdiagramm

Um den Prozess zu veranschaulichen, bietet sich das Aktivitätsdiagramm an. Es gibt einen Überblick des Verlaufs und kann als Grundlage dienen um mit allfälligen Stakeholdern sich sicher zu sein das man vom gleichen redet.

Ein Bild, das Text, Screenshot, Software, Website enthält.

Automatisch generierte Beschreibung

Aktivitätsdiagramm des Import-Prozesses

## User Stories

Um ein Verständnis der Anforderungen zu gewinnen, werden User Stories erfasst. Der Kern liegt in den Anforderungen der Berater und der Umsetzung der neuen Import-Funktion, jedoch habe ich mir die Freiheit genommen IPA relevante Kriterien ebenfalls in User Stories zu verpacken. So gewinne ich einen guten Überblick was zu erledigen ist und kann beim Testen gleich überprüfen ob auch die Entwicklerthemen richtig umgesetzt wurden.

### Rollen

Es gibt Versicherungsvertreter, Administratoren und Entwickler.

### Funktionale Anforderungen

|  |
| --- |
| **Story 1** |
| Als Versicherungsberater möchte ich meine Kundendaten aus dem alten "Angebot Software" System in das neue "OVWEB" System importieren können |
| **Akzeptanzkriterien** |
| **AC 1.1**: Der Berater kann einen Workflow im OVWEB starten, um die Daten zu importieren.  **AC 1.2**: Das System validiert das Excel-Format und importiert die Daten korrekt in die Customer-Tabellen von OVWEB.  **AC 1.3**: Das System verhindert den Import von Duplikaten, indem es vorhandene Kunden anhand von Nachnamen, Vorname und Geburtsdatum identifiziert und aktualisiert. |

|  |
| --- |
| **Story 2** |
| Als Versicherungsberater möchte ich eine Übersicht über den Import erhalten, um nachzuvollziehen was der Import Wizard gemacht hat. |
| **Akzeptanzkriterien** |
| **AC 2.1**: Nach dem Import wird eine Summary-Seite angezeigt, die dem Berater anzeigt, wie viele Kundendaten erfolgreich importiert wurden und wie viel aktualisiert wurden. |

|  |
| --- |
| **Story 3** |
| Als Versicherungsberater möchte ich in der OVWEB-Navigation auf die Import-Funktion zugreifen können. |
| **Akzeptanzkriterien** |
| **AC 3.1**: Die Import-Funktion ist in der Hauptnavigation von OVWEB integriert und leicht zugänglich. |

### Nichtfunktionale Anforderungen

|  |
| --- |
| **Story 4** |
| Als Versicherungsberater möchte ich, dass der Import-Wizard benutzerfreundlich ist, damit ich meine Daten selbstständig importieren kann. |
| **Akzeptanzkriterien** |
| **AC 4.1**: Der Import-Wizard verwendet eine klare und verständliche Sprache, die für nicht-technische Benutzer geeignet ist.  **AC 4.2**: Im Falle eines Fehlers während des Imports erhält der Berater klare Hinweise. |

|  |
| --- |
| **Story 5** |
| Als Entwickler möchte ich, dass alle neuen Funktionalitäten des OVWEB-Systems durch automatisierte Tests abgedeckt sind. |
| **Akzeptanzkriterien** |
| **AC 5.1**: Für die neue Funktion gibt es automatisierte Unit-Tests, die sowohl positive als auch negative Szenarien abdecken.  **AC 5.2**: Automatisierte Tests für null und non-null Datensätze sowie für leere und nicht-leere Kundenlisten sind implementiert und verifizieren die Handhabung dieser Fälle. |

|  |
| --- |
| **Story 6** |
| Als Administrator möchte ich, dass das System strukturiertes Logging implementiert, um die Nachverfolgung von Aktivitäten und die Fehlerbehebung zu erleichtern. |
| **Akzeptanzkriterien** |
| **AC 6.1**: Strukturiertes Logging ist durchgehend implementiert.  **AC 6.2**: Die Logs enthalten alle notwendigen Informationen, um die Aktionen nachvollziehen zu können.  **AC 6.3**: Persönlich identifizierbare Informationen (PII) werden nicht in den Logdateien ausgegeben. |

|  |
| --- |
| **Story 7** |
| Als Entwickler möchte ich, dass jede Anfrage mit einer Trace-ID nachverfolgt werden kann, um die Fehlersuche zu vereinfachen. |
| **Akzeptanzkriterien** |
| **AC 7.1**: Jede Anfrage im Frontend generiert eine eindeutige Trace-ID, die an das Backend weitergeleitet wird.  **AC 7.2**:Die Trace-ID wird konsistent über alle beteiligten Dienste hinweg weitergeleitet und in deren Logs festgehalten.  **AC 7.3**: Es ist möglich, ein Diagramm der Service-Aufrufe basierend auf den Trace-IDs zu erstellen. |

|  |
| --- |
| **Story 8** |
| Als Entwickler möchte ich, dass die UI-Komponenten des Systems wiederverwendbar sind. |
| **Akzeptanzkriterien** |
| **AC 8.1**: UI-Komponenten sind so strukturiert, dass komplexere Komponenten aus einfacheren zusammengesetzt sind.  **AC 8.2**: Komponenten sind im Storybook dokumentiert. |

# Planen

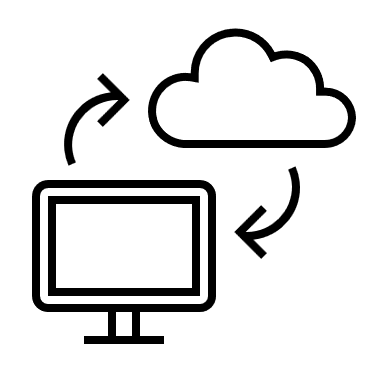
Das Planen ist die zweite Phase der IPERKA-Methode.

Hier werde ich aufzeigen, was die nötigen Schritte sind, um aus dem Ist-Zustand zum Soll-Zustand zu kommen.

## Soll-Zustand

### Überblick

Zuerst ein Überblick der nötigen Änderungen in den verschiedenen Komponenten der OVWEB-Applikationen.



Vereinfach besteht der Prozess aus drei Schritten: Einführung, Verarbeitung der Daten und Abschlussbericht. Bei einem Fehler wird der Berater aufgefordert die Generali zu kontaktieren.

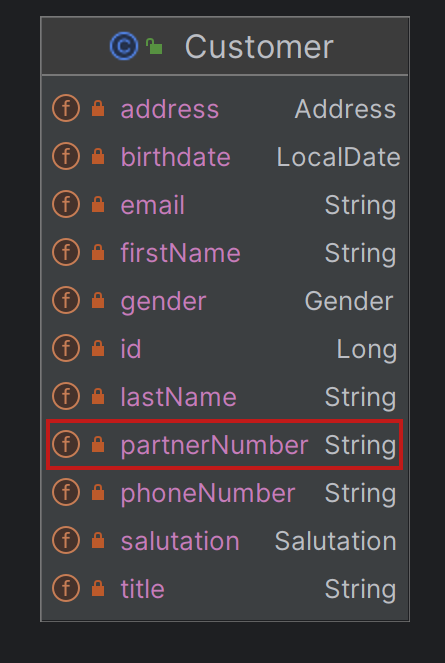
### Frontend

* Integrieren eines Links zum Import Wizard in die Hauptnavigation
* Erstellung einer Einführungsseite für den Import-Wizard, die den Prozess und die nächsten Schritte erklärt.
* Entwicklung einer Upload-Komponente, für das Hochladen von Excels.
* direktes Feedback über den Status der hochgeladenen Datei geben via Ladeanimation. Bei Fehlern -> Hinweis zur Kontaktaufnahme Generali AG
* Design und Implementierung einer Summary-Seite, die nach dem Import die Anzahl der neuen und aktualisierten Kundendaten anzeigt.
* Anzeigen von Fehlermeldungen

### Backend

* Services zur Verarbeitung des Imports und Speicherung der Daten in den Customer-Tabellen programmieren
* Validierung im Backend
* Logik für Erkennung bereits vorhandenen Kunden
* Aufbau strukturiertes Logging
* Implementierung von Distributed Tracing

### Klassendiagramm

Für die Implementierung des Import-Wizards und die damit Verbundenen Speicherung von Kundendaten aus der alten «ANGEBOT Software», ist eine Anpassung des Models «Customer» notwendig. Die Spalte «PartnerNo» in den zu Verfügung gestellten Excels ist nicht nur wichtige Information über den Kunden, sondern dient als Primärschlüssel um die beiden Excel-Sheets zu verbinden. Das Model im Backend soll um das Attribut «partnerNumber» erweitert werden, um die Identifikationsnummer des alten Tools speichern zu können.

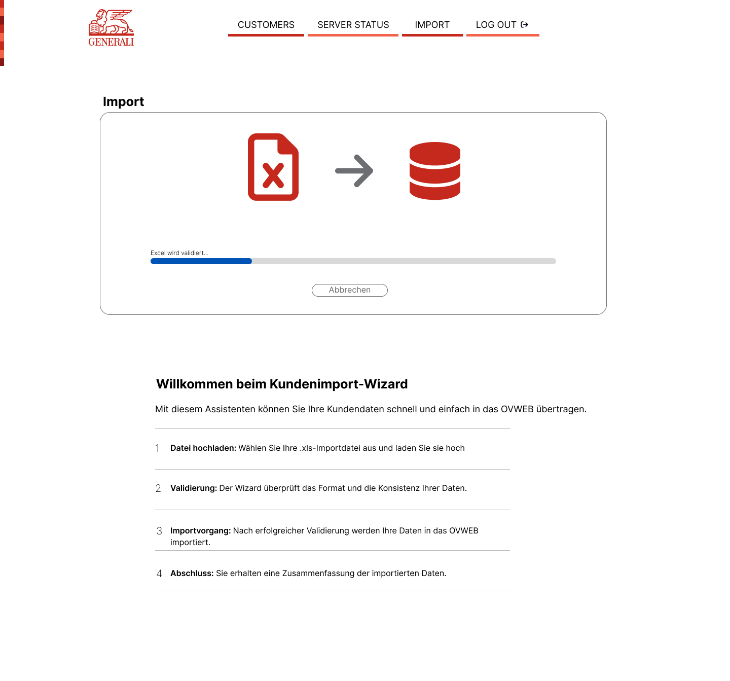
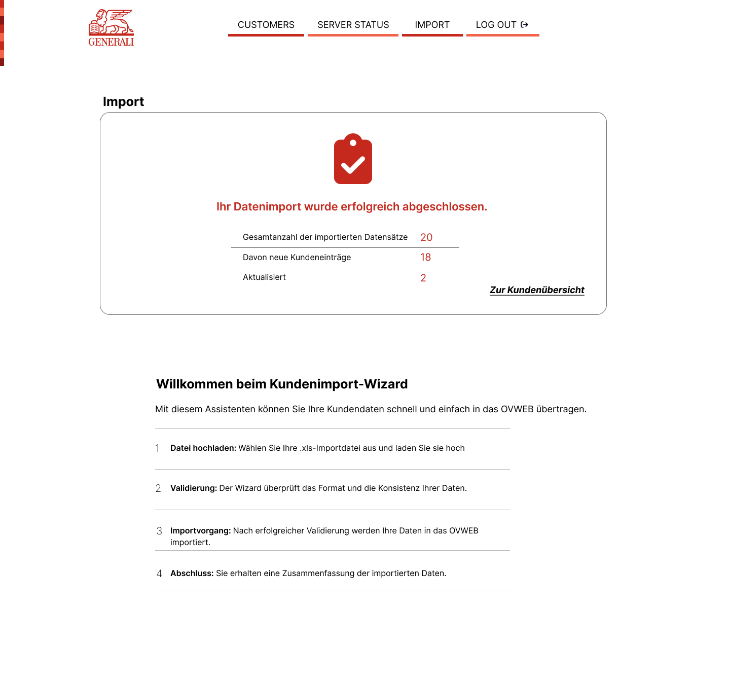
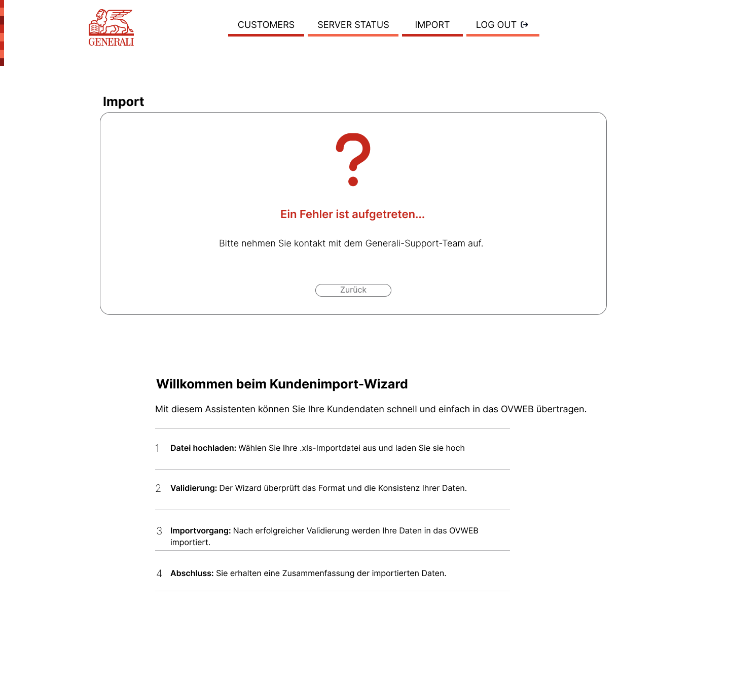
Erweiterung des «Customer» Models

### Datenbankanpassungen

Die Datenbank muss wie das Java Model in der «Customer» Tabelle angepasst werden. Die Spalte «partner\_number» wird via einer Flyway Migration dem Schema hinzugefügt.

## Mockups

Für die Planung des User Interfaces habe ich in Figma zwei Varianten umgesetzt. Ziel dieser Mockups ist es einen visuellen Überblick zu verschaffen was alles umzusetzen ist und die User Experience so gut wie möglich aufzuzeigen. Die Mockups bestehen aus den drei Seiten bzw. Schritte des Importwizards: Einführung, Verarbeitung und Abschluss. Zusätzlich eine Fehlerseite bei falschen Formaten oder Inkonsistente Daten.

Ein Bild, das Text, Screenshot, Schrift, Design enthält.

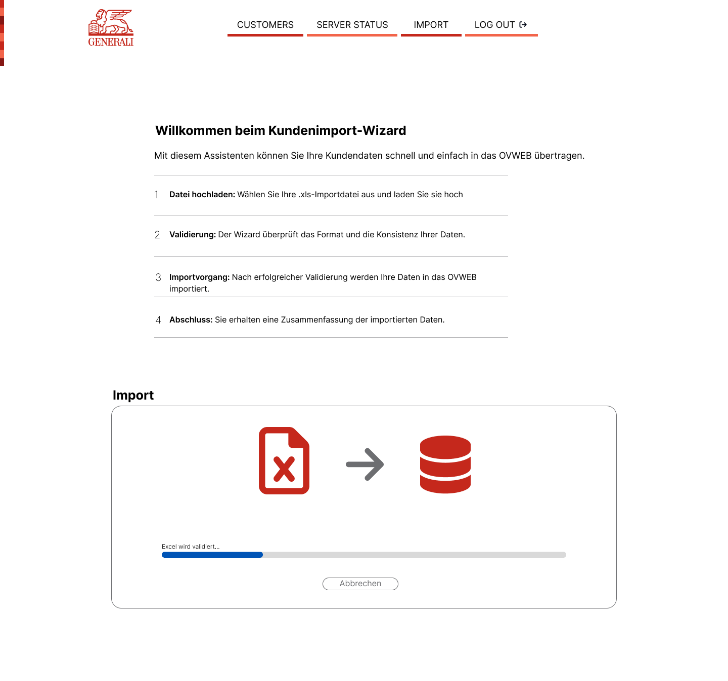
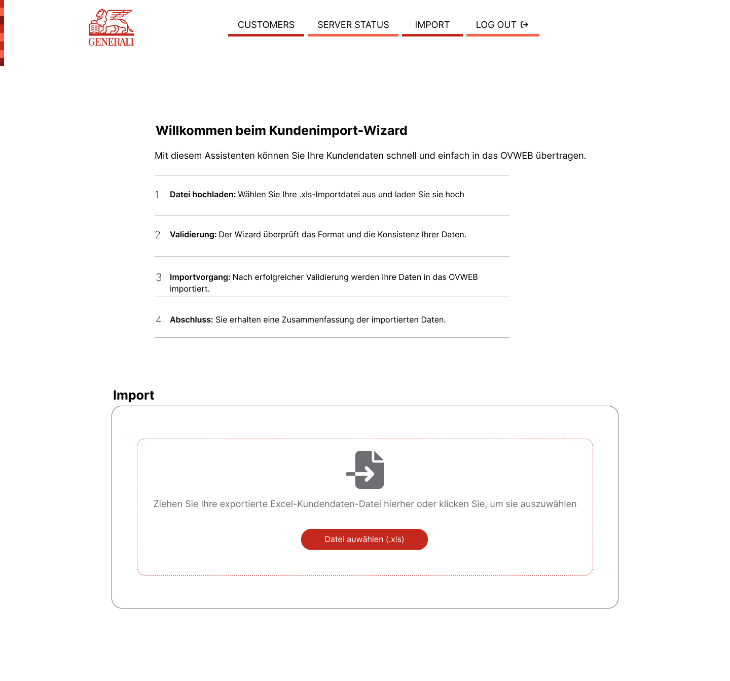
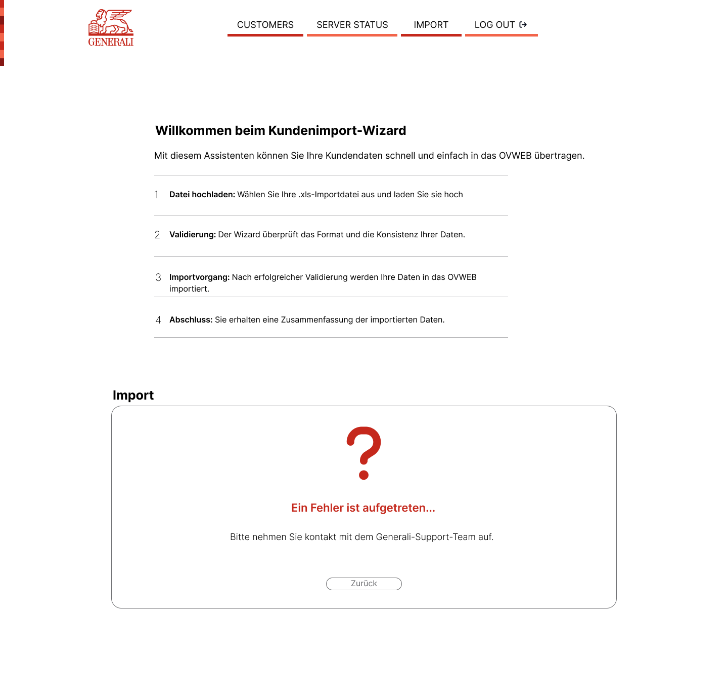
Automatisch generierte Beschreibung**Variante 1:**

Fehlerseite Variant 1

Zusammenfassung Variant 1

Verarbeitung Variant 1

Einführungsseite Variant 1

**Variante 2:**

Zusammenfassung Variant 2

Fehlerseite Variant 2

Verarbeitung Variant 2

Einführungsseite Variant 2

Die beiden Varianten Unterscheiden sich durch die Platzierung des Erklärtextes und des Import-Wizards. Bei der ersten Variante is das Import-Model zu oberst. Es legt Gewicht auf die Import Funktion und bietet dem Berater diese sofort an. Ist die Importseite bekannt und die Erklärung bei dem wiederholten Besuch der Seite ignoriert wird, bittet diese Variante das Wichtigste zuerst: Die Exportdatei auszuwählen und in das OVWEB einzulesen. Bei der zweiten Varianten wird der Berater bei jedem Besuch kurz durch den Prozess geführt. Am Ende der Einführung wartet der Importwizard am unteren Bereich der Seite. Bei Unsicherheiten muss der Berater nur kurz hinauf sehen bzw. scrollen um die Anleitung zu konsoltieren.

## Testkonzept

Es wird mit dem folgenden Testkonzept überprüft, ob die Arbeit korrekt umgesetzt wurde. In der fünften IPERKA Phase «Kontrollieren» werden die Testfälle dann getestet. Das Konzept baut stark auf den definierten Akzeptanzkriterien in der User Stories auf.

### Testziele und -strategie

* Ziel ist es, die Funktionalität, Validierung und Benutzerfreundlichkeit des Import Wizards zu testen.
* Ich verwende wo möglich Unit-Tests und decke die restlichen Kriterien mit manuelle Akzeptanztests ab.

### Testumgebung

Die Tests werden innerhalb der Generali AG mit einem Chrome Browser auf der lokalen Umgebung getestet. Die Springboot Applikation sowie das Next.Js Frontend wird gestartet und via Docker eine Instanz der neusten Version der Datenbanken und Keycloak hochgefahren. So baue ich eine Umgebung auf die so gut wie möglich eine Produktionsumgebung widerspiegelt.

### Testfälle

Das automatische Testen durch Unit-Tests sollten unter anderem folgende Punkte abdecken:

* Positive Testszenarien(Happy Path)
* Negative Testszenarien (falsche Formate/Inkonsistenz)
* Tests für null und non-null Skalar-Inputs sowie leere und nicht-leere Sammlungen (Collections)

sowie Story 5 und die dazugehörigen Akzeptanzkriterien.

Alle weiteren Anforderungen werden mit manuellen Akzeptanztests uberprüft.

Die folgenden Testfälle ergeben sich direkt aus den User Stories bzw. den Akzeptanzkriterien kurz AC.

|  |
| --- |
| **Testfall 1.1** |
| **Anforderung**: Story 1, AC 1.1  **Beschreibung**: Überprüfung, ob Berater Import-Workflow starten kann.  **Voraussetzung**: Benutzer ist im OVWEB-System als Versicherungsberater angemeldet.  **Eingabe:** Benutzer navigiert zum Import-Bereich |
| **Erwartetes Resultat** |
| Der Import-Workflow wird durch das Auswählen erfolgreich gestartet. |

|  |
| --- |
| **Testfall 1.2** |
| **Anforderung**: Story 1, AC 1.2  **Beschreibung**: Validierung der Excel-Datei und Import der Kundendaten.  **Voraussetzung**: Eine Excel-Datei im korrekten Format ist bereit zum Upload.  **Eingabe:** Hochladen der Excel-Datei durch den Benutzer. |
| **Erwartetes Resultat** |
| 1. Der Wizard erkennt das Format der Excel-Datei und bestätigt die Validierung. 2. Die Daten werden korrekt in die Customer-Tabellen von OVWEB importiert. |

|  |
| --- |
| **Testfall 1.3** |
| **Anforderung**: Story 1, AC 1.3  **Beschreibung**: Verhinderung des Imports von Duplikaten.  **Voraussetzung**: Eine Excel-Datei mit Daten, die Duplikate enthält.  **Eingabe:** Hochladen der Excel-Datei durch den Benutzer. |
| **Erwartetes Resultat** |
| Der Wizard erkennt die Duplikate und aktualisiert die Einträge |

|  |
| --- |
| **Testfall 2.1** |
| **Anforderung**: Story 2, AC 2.1  **Beschreibung**: Anzeige der Summary-Seite nach dem Import.  **Voraussetzung**: Erfolgreicher Abschluss des Importvorgangs.  **Eingabe:** - |
| **Erwartetes Resultat** |
| Die Summary-Seite zeigt die Anzahl der erfolgreich importierten und aktualisierten Kundendatensätze an. |

|  |
| --- |
| **Testfall 3.1** |
| **Anforderung**: Story 3, AC 3.1  **Beschreibung**: Zugänglichkeit der Import-Funktion in der Hauptnavigation.  **Voraussetzung**: Benutzer ist im OVWEB-System angemeldet.  **Eingabe:** - |
| **Erwartetes Resultat** |
| Die Import-Funktion ist sichtbar und kann von der Hauptnavigation aus erreicht werden. |

|  |
| --- |
| **Testfall 3.1** |
| **Anforderung**: Story 3, AC 3.1  **Beschreibung**: Zugänglichkeit der Import-Funktion in der Hauptnavigation.  **Voraussetzung**: Benutzer ist im OVWEB-System angemeldet.  **Eingabe:** - |
| **Erwartetes Resultat** |
| Die Import-Funktion ist sichtbar und kann von der Hauptnavigation aus erreicht werden. |

|  |
| --- |
| **Testfall 4.1-4.2** |
| **Anforderung**: Story 4, AC 4.1-4.2  **Beschreibung**: Benutzerfreundlichkeit und Fehlerbehandlung im Import-Wizard.  **Voraussetzung**: Mehreres Ausführen des Imports  **Eingabe:** Korrekte und inkorrekte Excel-Dateien werden vom Benutzer hochgeladen. |
| **Erwartetes Resultat** |
| Der Wizard verwendet eine klare Sprache und gibt bei Fehlern klare Hinweise. |

|  |
| --- |
| **Testfall 6.1-6.2** |
| **Anforderung**: Story 6, AC 6.1-6.2  **Beschreibung**: Überprüfung des strukturierten Loggings.  **Voraussetzung**: Importvorgang wurde durchgeführt.  **Eingabe:** - |
| **Erwartetes Resultat** |
| Logs sind strukturiert und enthalten alle notwendigen Informationen ohne PII. |

|  |
| --- |
| **Testfall 7.1-7.3** |
| **Anforderung**: Story 7, AC 7.1-7.3  **Beschreibung**: Nachverfolgbarkeit der Anfragen mittels Trace-ID.  **Voraussetzung**: Mehrere Anfragen wurden vom Frontend initiiert.  **Eingabe:** - |
| **Erwartetes Resultat** |
| 1. Jede Anfrage hat eine eindeutige Trace-ID, die in allen beteiligten Services protokolliert ist. 2. Ein Diagramm der Service-Aufrufe kann erstellt werden. |

|  |
| --- |
| **Testfall 8.1-8.3** |
| **Anforderung**: Story 8, AC 8.1-8.2  **Beschreibung**: Wiederverwendbarkeit der UI-Komponenten.  **Voraussetzung**: UI-Komponenten sind vorhanden.  **Eingabe:** - |
| **Erwartetes Resultat** |
| Komplexere Komponenten setzen sich aus einfacheren zusammen und sind im Storybook dokumentiert. |

# Entscheiden

Im Anschluss der Planung gilt es sich für eine Lösungsvariante zu Entscheiden. Es wurden zwei Variationen als Mockups im Figma umgesetzt. In diesem Kapitel möchte ich die beiden Optionen vergleichen und eine definitive Entscheidung treffen, welche umgesetzt wird. So kann ich mit einem klaren Bild in die Realisierung übergehen.

## Ein Bild, das Text, Screenshot, Schrift, Design enthält. Automatisch generierte BeschreibungEin Bild, das Text, Screenshot, Schrift, Design enthält. Automatisch generierte BeschreibungLösungsvarianten im Vergleich

Einführungsseite Variant 1

Einführungsseite Variant 2

Hier sind die beiden Einführungsseiten nochmals zur Veranschaulichung.

Die Versionen sind sich ähnlich. Der Unterscheid liegt in der Gewichtung des Import-Wizards auf der Seite. Soll er die grösste Gewichtung haben, noch mehr als der Info-Text? Bei der ersten Variante ist der Import-Wizard zuoberst. Ich sehe das Argument das ein versierter Benutzer sofort von der Funktion gebrauch machen will und nach mehreren Imports die Erklärung überspringt. Jedoch ist einer der zentralsten Anforderungen die Benutzerfreundlichkeit und das selbständige Verwenden ohne technischen Hintergrund. So bietet sich die zweite Version an. Zuerst die Erklärung und dann die Anwendung. Auch während dem Verwenden des Wizard kann der Berater so die Anleitung nochmals kurz konsultieren mit einer kurzen blick nach oben.

Die Wahl fällt somit auf die Variante 2, da sie den Benutzer sofort bei der Hand nimmt und ihn durch die Applikation führt.

# Realisierung

## Anpassungen in der Datenbank und Model

Um die Daten richtig persistieren zu können, sind Anpassungen in der

Datenbank und im Model nötig.

Mit Hilfe von Flyway wurden Datenbankanpassungen an der Tabelle **«customer»** vorgenommen. Es wurde eine neue Spalte hinzugefügt: **«partner\_number»** vom Typ varchar(255).

Hierfür wurde die Datei **«V1\_0\_10\_\_add\_partner\_number\_to\_customertabel.sql»** erstellt.Flyway erleichtert das Versionsmanagement der Datenbankstruktur, indem es SQL-Skripte wie diese auf automatisierte Weise während des Hochfahrens der Applikation ausführt.

Wie in der Planung erwähnt, wurde das Attribut «partnerNumber» dem «Customer» Model hinzugefügt.

## Erstellen eines Import Services

Ich entscheide mich einen neuen Import Service anzulegen. Der Code ist so getrennt von den CRUD-Operationen des Customer Services. Für das Verarbeiten der Excel Daten verwende ich die Apache-POI Libraries. Apache bietet diverse Klassen und Methoden, um Excels und ähnliche Spreadsheets zu verarbeiten. Auch wenn dies nicht in den Anforderungen ist, schrieb ich zuerst eine Hilfsmethode, die zwischen .xls und .xlsx, dem neuern auf XML basierendem Excel-Format, unterscheidet und sie richtig verarbeitet. Während der Implementierung des Services zum Lesen und Speichern der Kundendaten in den Excel-Dateien, arbeitete ich eng mit den mir zur Verfügung gestellten Testdaten. Bei den numerischen Formaten wie zum Beispiel der Postleitzahl, entschied ich mich die Datentypen vorher zu überprüfen, anstatt einfach anzunehmen das die Excels mir immer konsistente und richtige Datentypen liefern. So kann der Service auch bei String Werten in diesen Feldern, die Daten verarbeiten.

Die Hauptmethode der Klasse **«ImportService.java»** ist die **importData**-Methode. Sie gibt eine Liste aller erstellten Kunden zurück, bei Fehlern eine Leere Liste zurück. Sie ist mit einer Annotation als transaktional deklariert worden, damit sie auch bei Fehlern auf der aller letzten Zeile der importierten Excel alle Speicherung von Kundendaten wieder rückgängig macht.

Die Methode **«saveOrUpdateCustomer»** ist für nicht nur für das richtige Speichern zuständig, sondern auch für Kontrolle, ob ein Kunde bereits in unserer Datenbank existiert. Dank JPA-Repository kann ich im Interface **«CustomerRepository.java»** bequem eine neuen Methode **«findByLastNameAndFirstNameAndBirthdate»** schreiben und sie verwenden um zu Überprüfen ob ein Datensatz neu angelegt werden muss oder aktualisiert werden muss.

// Mit Grafiken arbeiten für abstrakte Konzepte

## zz

## Code Beispiel:

const validateAndSetError = (value, originalSuggestions, form, fieldName) => {

        if (!originalSuggestions.some((suggestion) => suggestion.Value === value)) {

            form.setError(fieldName, {

                message: 'Please choose a suggestion'

            });

        } else {

            form.clearErrors(fieldName);

        }

    };

## Datenbank blah blah

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

# Kontrollieren

Das Kontrollieren ist die fünfte Phase der IPERKA-Methode.

## Testing

### Testkonzept

### Testprotokoll

Angaben PC & Software:

|  |  |
| --- | --- |
| **Name** | Lenovo ThinkBook 15 G4 ABA |
| **Prozessor** | AMD Ryzen 7 5825U, 2000 MHz, 8 Kerne, 16 logische Prozessor |
| **Ram** | 16 GB |
| **Testumfeld / Browser** | Google Chrome Version 119.0.6045.124 (Offizieller Build) (64-Bit) |
| **Betriebssystem** | Microsoft Windows 11 Pro |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Testfall** | **Resultat** | **Datum** | **Tester** | **Bemerkung** |
| **1** | Passed | 15.11.2023 | J.Baars |  |
| **2** | Failed |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

### Testbericht

Bemerkungen zu Tests:

## Unittests

# Auswerten

## Vergleich Ist / Soll

## Persönliches Fazit zur IPA

### Was war gut?

### Was lief weniger gut?

## Schlussreflexion

# Glossar

|  |  |
| --- | --- |
| **Begriff** | **Erklärung** |
|  |  |
|  |  |
|  |  |

# Quellenverzeichnis

|  |
| --- |
| Wikipedia: Aktivitätsdiagramm  https://de.wikipedia.org/wiki/Aktivit%C3%A4tsdiagramm  [Abrufdatum: 06.02.2024]  Bemerkung: Richtige Notation und Orientierung für Akitvitätsdiagram in der Planung |
| Thomas Gabathuler: Was ist IPERKA  <https://www.tgabathuler.ch/Iperka>  [Abrufdatum: 08.02.2024]  Bemerkung: Im Kapitel «IPERKA-Methode» verwendet für Wording und Umfang |
| swagger: File Upload  <https://swagger.io/docs/specification/2-0/file-upload/>  [Abrufdatum: 23.10.2023]  Bemerkung: Verwendet in der Umsetzung des Import |
| geeksforgeeks: Apache POI | Getting Started  https://www.geeksforgeeks.org/apache-poi-getting-started/  [Abrufdatum: 08.02.2024]  Bemerkung: Verwendet für Wording in der Dokumentation und Umsetzung des Import Services |

## Einsatz von generativen Tools

User Stories

Testkonzept

Bewerten ob alles abgedeckt

Generieren von Swagger dokumentation

# Anhang

## Sitzungsprotokolle

### Erster Expertenbesuch

Der erste Expertenbesuch fand via Google Meet statt. Herr Kaufmann moderierte das Meeting und wir stellten uns alle vor. Nachdem er die verschiedenen Rollen während der IPA erklärt hat, vergewisserte er sich das wir alle die Aufgabenstellung gleich verstanden haben. Im Anschluss stellte er ein paar detailierte Fragen zu gewissen Themen via z.B die Rendering Art, die ich vor habe du verwenden bei den Next.Js Seiten die ich implementieren muss.

Ich konnte meine Fragen zum Umfang des Quellenverzeichnis klären. Danach schauten wir gemeinsam den von mir erstellten Zeitplan für die IPA an. Er wies stark darauf hin, dass die von mir rund 24 geplanten Arbeitsstunden für die Dokumentation zu wenig sind, um alle nötigen Punkte des IPA-Berichts abzudecken. Ich versuchte aufzuzeigen das diverse Aufgaben auch auf dem Konto der Dokumentationsarbeiten verbucht werden konnten und diese Zahl dann ansteigen würde, jedoch nahm ich sein Feedback dankend an und fing bereits an zu überlegen, wie ich das besser in meinem Zeitplan hervorheben könnte. Herr Kaufmann betonte nochmals, dass die Dokumentation die grösste Gewichtung im Kriterienkatalog findet. Termine für die zweiten Expertenbesuch und das Fachgespräch wurden bestätigt bzw. abgemacht. So beendeten wir das Meeting für Heute. Ich diskutierte im Anschluss noch mit Roman ein wenig über diverse Punkte des Kriterienkatalogs.

### Zweiter Expertenbesuch