**IPA-Dokumentation**

**Erweiterung des Competec Lernportals**

Ein Bild, das Text, Whiteboard, Handschrift, Im Haus enthält.

Automatisch generierte Beschreibung

Inhaltsverzeichnis

[Allgemeine Informationen 4](#_Toc160533540)

[Versionierung Dokumentation 4](#_Toc160533541)

[Versionierung Praktische Arbeit 4](#_Toc160533542)

[Zusatzinformationen 4](#_Toc160533543)

[Teil 1 – Umfeld und Ablauf 5](#_Toc160533544)

[Detaillierte Aufgabenstellung 5](#_Toc160533545)

[Titel der Arbeit 5](#_Toc160533546)

[Ausgangslage 5](#_Toc160533547)

[Teilauftrag 1: Login und Benutzerverwaltung 6](#_Toc160533548)

[Teilauftrag 2: Bilderverwaltung 7](#_Toc160533549)

[Teilauftrag 3: Fehlerbehandlung des Datenimports 7](#_Toc160533550)

[Mittel und Methoden 8](#_Toc160533551)

[Deklaration der Vorkenntnisse 9](#_Toc160533552)

[Deklaration der Vorarbeiten 9](#_Toc160533553)

[Neue Lerninhalte 10](#_Toc160533554)

[Arbeiten in den letzten 6 Monaten 10](#_Toc160533555)

[Deklaration der benutzten Firmenstandards 10](#_Toc160533556)

[Projektaufbauorganisation 11](#_Toc160533557)

[Projektmethode IPERKA 12](#_Toc160533558)

[Einfluss konkret auf meine IPA: 12](#_Toc160533559)

[Zeitplan 13](#_Toc160533560)

[Arbeitspakete 14](#_Toc160533561)

[Meilensteine 17](#_Toc160533562)

[Organisation der Arbeitsergebnisse 18](#_Toc160533563)

[Versionierung 18](#_Toc160533564)

[Backup 18](#_Toc160533565)

[Quellcode / Skripts 18](#_Toc160533566)

[Arbeitsplatz 18](#_Toc160533567)

[Arbeitsjournal 19](#_Toc160533568)

[Tag 1 – 29.02.2024 19](#_Toc160533569)

[Tag 2 – 04.03.2024 20](#_Toc160533570)

[Tag 3 – 05.03.2024 21](#_Toc160533571)

[Tag 4 – 06.03.2024 22](#_Toc160533572)

[Tag 5 – 07.03.2024 23](#_Toc160533573)

[Tag 6 – 11.03.2024 24](#_Toc160533574)

[Tag 7 – 12.03.2024 25](#_Toc160533575)

[Tag 8 – 13.03.2024 26](#_Toc160533576)

[Tag 9 – 14.03.2024 27](#_Toc160533577)

[Tag 10 – 18.03.2024 28](#_Toc160533578)

[Teil 2 – Projekt 29](#_Toc160533579)

[Kurzfassung 29](#_Toc160533580)

[Ausgangslage 29](#_Toc160533581)

[Umsetzung 29](#_Toc160533582)

[Ergebnis 29](#_Toc160533583)

[Phase "Informieren" 30](#_Toc160533584)

[Strategien zur Fehlerbehandlung 30](#_Toc160533585)

[Informationsbeschaffung Bilderverwaltung 32](#_Toc160533586)

[Phase "Planen" 34](#_Toc160533587)

[Ist-Zustand des Systems 34](#_Toc160533588)

[Soll-Zustand des Systems 36](#_Toc160533589)

[Datenbankmodell 38](#_Toc160533590)

[Tabellenstruktur 40](#_Toc160533591)

[Testkonzept 43](#_Toc160533592)

[Phase "Entscheiden" 45](#_Toc160533593)

[Einleitung 45](#_Toc160533594)

[Fehlerbehandlung 45](#_Toc160533595)

[Bilderverwaltung 45](#_Toc160533596)

[Benutzerverwaltung 45](#_Toc160533597)

[Sessionverwaltung 45](#_Toc160533598)

[Image Tabelle 46](#_Toc160533599)

[Importzeitpunkte 46](#_Toc160533600)

[Fazit 46](#_Toc160533601)

[Phase "Realisieren" 47](#_Toc160533602)

[Realisierungskonzept 47](#_Toc160533603)

[Arbeitspaket 13 48](#_Toc160533604)

[Arbeitspaket 14 50](#_Toc160533605)

[Phase "Kontrollieren" 65](#_Toc160533606)

[Phase "Auswerten" 66](#_Toc160533607)

[Persönliches Fazit 67](#_Toc160533608)

[Quellenverzeichnis 68](#_Toc160533609)

[Glossar 69](#_Toc160533610)

# 

# Allgemeine Informationen

## Versionierung Dokumentation

|  |  |  |
| --- | --- | --- |
| Version | Datum | Branch |
| 1 | 29.02.2024 | https://github.com/2David4/IPA-Dokumentation/tree/ipa-day1 |
| 2 | 04.03.2024 |  |
| 3 | 05.03.2024 |  |
| 4 | 06.03.2024 |  |
| 5 | 07.03.2024 |  |
| 6 | 11.03.2024 |  |
| 7 | 12.03.2024 |  |
| 8 | 13.03.2024 |  |
| 9 | 14.02.2024 |  |
| 10 | 18.03.2024 |  |

## Versionierung Praktische Arbeit

|  |  |
| --- | --- |
| Projekt | Branch |
| Competec/ita-talent-api | https://github.com/Competec/ita-talent-api/tree/ipa-du |
| Competec/ita-talent-client | https://github.com/Competec/ita-talent-client/tree/ipa-du |

## Zusatzinformationen

|  |  |
| --- | --- |
| Dokumenttitel | Dokumentation |
| Autor | David Unterguggenberger |
| Dateiname |  |
| Ablageort | Lokaler Speicher, Github, Google Drive |

# Teil 1 – Umfeld und Ablauf

## Detaillierte Aufgabenstellung

### Titel der Arbeit

Administrative Verwaltung des CompAcademy Lernportals (Personalentwicklung)

### Ausgangslage

Die Competec-Gruppe besteht aus den Handelsunternehmen Alltron AG, Jamei AG, Medidor AG, Schoch Vögtli AG und BRACK.CH AG, dem Logistikdienstleister Competec Logistik AG und der Competec Service AG, die alle zentralen Dienste übernimmt. Die Firmengruppe mit Hauptsitz in Mägenwil AG beschäftigt rund 1400 Mitarbeitende und bildet 77 Lernende in diversen Berufsbildern aus. Die Grundausbildung der IT-Lernenden geschieht in der internen IT-Academy. Als Teil der Ausbildung werden verschiedenen Applikationen für unsere Firmengruppe entwickelt. Die Lernenden können dabei von der Entwicklung über die Projektleitung bis zur Inbetriebnahme verschiedene Rollen in der Softwareentwicklung einnehmen.

Die interne Personalentwicklung (im weiteren als "CompAcademy" bezeichnet) schreibt Kursangebote aus, auf welche sich die Mitarbeiter der Competec anmelden können. Die Verwaltung der Kurse geschieht auf der Talent Management Lösung von Umantis AG (im weiteren als "LMS" bezeichnet). Aufgrund der Benutzerunfreundlichkeit des LMS wurde die IT-Academy beauftragt eine visuell ansprechendere Lösung zu entwickeln. Daraus entstand die Plattform "CompAcademy Lernportal" (im weiteren als "Lernportal" bezeichnet), welche die Daten des LMS in regelmässigen Zeitabständen abfragt und benutzerfreundlich aufbereitet. Aufgrund von organisatorischen Schwierigkeiten wurde eine vereinfachte Variante der Anforderungen produktiv aufgeschaltet. Nun sollen die verbleibenden Features in einem Update umgesetzt werden.

Aktueller Stand

==============

Das Lernportal besteht aus einem Java Backend, welches mittels Spring Boot implementiert wurde. Dazu wurde das UI mit React implementiert, welches statisch bereitgestellt wird. Die Kommunikation zwischen den Systemen wird mit einer Websocket Verbindung sichergestellt.

Im Lernportal wurden bereits folgende Funktionalitäten implementiert.

Automatischer Datenimport

-------------------------------------

Im LMS wurde ein automatischer Export erstellt, welcher alle Kurse zur Verfügung stellt. Dieser funktioniert über einen REST-Endpunkt, welcher die Daten im XML-Format zurückgibt.

Diese Schnittstelle wird vom Lernportal im 10 Minuten Takt angefragt. Die Kursdaten werden anschliessend konvertiert, normalisiert und in einer PostgreSQL Datenbank abgespeichert. Gleichzeitig wird aus dem Titel und der Beschreibung der Kurse ein Suchindex erstellt.

Webseite / UI

------------------

Die Website besteht aus den folgenden Ansichten

• Suchresultat Seite

• Kurskatalog aufgeteilt auf Kurskategorien (Für alle Mitarbeiter, Für deine Fachrolle, Für alle Führungskräfte) mit Filtermöglichkeiten nach Typ, Organisation und Unterkategorien

• Detaillierte Kursansicht mit den Detailinformationen, Anmeldung und Teilveranstaltungen.

Mockup

-----------

Das Mockup der gesamten Webapplikation, einschliesslich der IPA-Erweiterungen, wurde bereits mit Figma erstellt. Dabei bildet es einen Richtwert und muss nicht pixelgenau übertragen werden.

Detaillierte Aufgabenstellung

Die Erweiterung des Lernportals ist in drei Teilaufträge gegliedert. Das Deployment ist nicht teil der IPA und muss nur lokal lauffähig sein.

### Teilauftrag 1: Login und Benutzerverwaltung

=======================================

Das Lernportal soll mit einem geschützten Administrationsbereich erweitert werden. Auf diesen kann nur mit einer gültigen Username- und Passwortkombination zugegriffen werden. Zusätzlich soll dafür eine Benutzerverwaltung erstellt werden.

Anforderungen

---------------------

• Ein Benutzer besteht mindestens aus den folgenden Datenpunkten

-- Benutzername

-- E-Mail

-- Passwort

-- Aktiv / inaktiv

• Der Benutzername muss eindeutig sein.

• E-Mails müssen validiert werden.

• Passwörter müssen mindestens 4 Zeichen lang sein.

• Deaktivierte Benutzer können sich nicht einloggen.

• Nach erfolgreichem Login soll der Benutzer 24 Stunden Zugriff auf die Applikation haben, bevor dieser sich erneut anmelden muss.

• Angemeldete Benutzer können sich ausloggen.

• Die administrativen Navigationselemente sind nur für eingeloggte Benutzer ersichtlich

• Die Benutzerverwaltung ermöglicht das Erstellen, Bearbeiten und Deaktivieren von Benutzer.

• Alle angemeldeten Benutzer haben die gleichen Berechtigungen

• Angemeldete Benutzer können alle Benutzer ohne Einschränkung bearbeiten (Es ist ein internes Tool ohne Hierarchie / Rollenkonzept ausser "Angemeldet oder nicht")

### Teilauftrag 2: Bilderverwaltung

===========================

Um die Kursangebote attraktiver zu gestalten, sollen die Kurse mit einem Bild versehen werden. Da im LMS keine Bilder erfasst werden können, muss die Verwaltung im Lernportal implementiert werden.

Anforderungen

---------------------

• Ein Bild besteht mindestens aus den folgenden Datenpunkten

-- Bild

-- Beschreibung

-- Kurs ID

• Die Bilderverwaltung ermöglicht das Erstellen, Bearbeiten und Löschen von Kursbildern.

• Wenn ein Kursbild vorhanden ist, wird dieses im Kurskatalog und der detaillierten Kursansicht dargestellt.

• Die Bilderverwaltung ist nur für eingeloggte Benutzer ersichtlich.

• Die Assoziation zwischen Bild und Kurs geschieht über die Kurs ID

• Bilder welche breiter als 1024px sind werden auf 1024px herunterskaliert.

• Folgende Bildformate werden unterstützt: PNG, JPG

• Bilder werden über eine REST-Schnittstelle angefragt, hochgeladen, angepasst und gelöscht.

• Die Bilder werden in der Datenbank abgespeichert.

• Die Bilddaten werden unabhängig von den Daten des Imports gespeichert.

• Wenn kein Kursbild vorhanden ist, wird nichts angezeigt.

### Teilauftrag 3: Fehlerbehandlung des Datenimports

=============================================

Beim Datenimport vom LMS in das Lernportal werden die Daten aufbereitet und normalisiert. Transformierungen werden aktuell nur in die Logdateien geschrieben. Damit diese Datenveränderungen nachvollzogen werden können, sollen diese den Administratoren sichtbar gemacht werden. Dadurch können Administratoren Erfassungsfehler im LMS-Quellsystem korrigieren.

Diese Transformierungen können diverse Ursachen haben. Dabei sind aktuell folgende bekannt:

• Fehlende Daten bei einem Pflichtfeld

-- Beispiel: Kurs hat keine Kategorie

-- Resultat: Standartwert wird angenommen

• Daten können nicht konvertiert werden

-- Beispiel: String anstatt Zahl

-- Resultat: Datenpunkt wird ignoriert

• Kurs referenziert einen nicht exportierten Kurs

-- Beispiel: Referenzierter Kurs wurde historisiert

-- Resultat: Kurs wird ignoriert

Zusätzlich soll bei jeden Datenimport der Zeitpunkt und Ausführungsstatus festgehalten werden.

Anforderungen

---------------------

• Transformierung des Datenimports werden in der Datenbank festgehalten.

• Eine Transformierung besteht mindestens aus den folgenden Datenpunkten

-- Kurs ID

-- Typ (Kritisch, Konvertierung, fehlende Daten)

-- Grund / Ursprung

-- Resultat

• Für jeden Datenimport werden die alten Transformationseinträge gelöscht

• Bei jeden Datenimport wird der Zeitpunkt und Status in der Datenbank festgehalten

• Ein Datenimport besteht mindestens aus den folgenden Datenpunkten

-- Zeitpunkt

-- Status (OK, Netzwerkfehler, Genereller Fehler)

• Die letzten 10 Datenimporteinträge sollen angezeigt werden.

### Mittel und Methoden

Versionsverwaltung

--------------------------

Der Quellcode soll auf dem Competec GitHub versioniert werden.

Dabei sind folgende Repositorien zu benutzen:

• ita\_talent\_api

• ita\_talent\_client

Code / Projektrichtlinien

--------------------------------

Der bestehende Projektaufbau soll übernommen und dem Auftrag entsprechend erweitert werden.

Style Guidelines

---------------------

Intellij Google Java Style Guide | https://github.com/google/styleguide/blob/gh-pages/intellij-java-google-style.xml

Teststrategie

-----------------

Um eine korrekte Ausführung zu gewährleisten, sollten alle Anforderungen manuell getestet werden. Dabei müssen alle Anforderungen validiert und abgedeckt werden. Dabei sind keine automatisierten Tests gefordert.

Umgebung

----------------

Firmen-Notebook, Windows 10, IntelliJ IDEA Ultimate, Figma, Git, PostgreSQL, Java 17, weitere Tools nach Bedarf

Technologien

------------------

Java 17, JavaScript, PostgreSQL, Spring Boot 3.1.x, React 18.2.x

Mockup

-----------

Figma Projekt im IT-Academy Team.

### Deklaration der Vorkenntnisse

In den vergangenen Monaten hat der Auszubildende folgende Vorkenntnisse erarbeitet.

• Backend

-- Java

-- Spring Boot

• Frontend

-- TypeScript

-- React

• Datenbank

-- PostgreSQL

• Protokoll / Format

-- Rest

-- XML

-- JSON

### Deklaration der Vorarbeiten

In einem Probeprojekt die Kenntnisse im Datenaustausch mit Websockets und programmatische Bildbearbeitung in Java vertiefen.

Das GitHub Projekt ist in einem sauberen Zustand und ein aktueller Branch für die IPA wurde erstellt.

### Neue Lerninhalte

Der Auszubildende wird keine neuen Lerninhalte erarbeiten.

### Arbeiten in den letzten 6 Monaten

IT-Academy Community

--------------------------------

Der Auszubildende war massgebend an dem Projekt "IT-Academy Community" beteiligt. Dabei handelt es sich um eine Webplattform bestehend aus einem React Frontend und Spring Boot Backend. In diesem Projekt arbeitete der Auszubildende an der Planung, Datenbankentwurf, Backend, Frontend und Deployment.

Tools/

-------

Das Projekt "Tools/" ist eine Webseite, welche alle aktiv intern genutzten Applikationen übersichtlich darstellt. Der Auszubildende übernahm in diesem Projekt den Lead und arbeitete an der Planung, Datenerhebung, Frontend und Deployment der Applikation.

Testprojekt

---------------

Zur Vorbereitung auf die IPA erstellte der Auszubildende ein Testprojekt. Dabei konnten alle fehlenden Kenntnisse erarbeitet werden. Um ein tiefes Verständnis zu erarbeiten, wurde die initiale Erstellung, Datenbankentwurf, Backend und Frontend von dem Auszubildenden umgesetzt.

## Deklaration der benutzten Firmenstandards

Im Rahmen meiner Individuellen Praktischen Arbeit (IPA) bei der Competec AG habe ich mich streng an die internen Firmenstandards gehalten, um sowohl die Konsistenz als auch die professionelle Qualität meiner Arbeit sicherzustellen.

Für die Erstellung der Dokumentation habe ich die unternehmenseigene Word-Vorlage genutzt. Diese Vorlage ist speziell darauf ausgelegt, die Lesbarkeit und Übersichtlichkeit von Berichten und Dokumenten zu optimieren und gleichzeitig das professionelle Erscheinungsbild von Competec zu wahren.

| Rolle | Name | Aufgabe und Verantwortung |
| --- | --- | --- |
| Kandidatin,  Kandidat | David Unterguggenberger | Die Anforderungen in der Aufgabenstellung  werden innerhalb der gegebenen Zeit vollständig  und korrekt erfüllt und umgesetzt. Die volle  Verantwortung wird übernommen |
| Berufsbildnerin,  Berufsbildner | Christian Walder | Kann verschiedene, unterstützende und administrative Aufgaben übernehmen, wertet jedoch die IPA-Arbeit nicht aus. |
| Verantwortliche Fachkraft | Flo Eberle | Entwickelt die Aufgabenstellung. Garantiert, dass die KAND während der IPA ungestört arbeiten können und beurteilt die erzielten Resultate der Arbeit. Ist verantwortlich für einen reibungslosen Ablauf der IPA |
| Hauptexpertin,  Hauptexperte | Jan Hohenheim | Besucht die Lehrfirma drei Mal und beurteilt die erbrachte Leistung und Qualität der Arbeit. Er ist für eine korrekte Umsetzung und Bewertung der IPA verantwortlich. Die Fachgespräche können auf Grund der Corona Pandemie auch vom Homeoffice aus durchgeführt werden. |
| Nebenexperte,  Nebenexpertin | Sara Zarubica | Ist für die korrekte Protokollierung der Präsentation, der Demo und des Fachgesprächs |

## Projektaufbauorganisation

## Projektmethode IPERKA

Die IPERKA-Methode ist ein strukturierter Ansatz für die Planung und Durchführung von Projekten. Die Abkürzung IPERKA steht für die sechs Phasen des Projektmanagements: Informieren, Planen, Entscheiden, Realisieren, Kontrollieren und Auswerten. Dieser methodische Rahmen bietet eine klare Richtlinie für die systematische Bearbeitung und Reflexion von Projektaufgaben.

### Einfluss konkret auf meine IPA:

|  |  |
| --- | --- |
| Phase | Aktivitäten |
| Informieren | Recherche über bestehende Lösungen für Login- und Benutzerverwaltungssysteme, Bilderverwaltung und Fehlerbehandlung bei Datenimporten. Analyse der Anforderungen und der Bewertungskriterien. |
| Planen | Entwicklung eines Konzepts für die Umsetzung der Aufgabenstellung, einschliesslich Datenbankmodell. Festlegung von Meilensteinen und Arbeitspakete. |
| Entscheiden | Festlegung der Architektur und Struktur der Datenbank, Auswahl spezifischer Methoden für die Fehlerbehandlung im Datenimport sowie Entscheidungen bezüglich der Implementierungsdetails für Login und Benutzerverwaltung, unter Berücksichtigung der Effizienz und Sicherheit. |
| Realisieren | Jetzt wird das geplante in die Realität umgesetzt! Implementierung der Benutzerverwaltung, einschliesslich Login-Funktionalität und Sicherheitsmassnahmen. Entwicklung der Bilderverwaltungsfunktionalität und Integration in das bestehende Lernportal. Umsetzung der Fehlerbehandlung für den Datenimport. |
| Kontrollieren | Durchführung von Tests zur Überprüfung der Funktionalitäten und Sicherheit der implementierten Features. Überwachung der Leistung und Anpassung basierend auf Testergebnissen. |
| Auswerten | Bewertung des Projektfortschritts und der erreichten Ergebnisse im Vergleich zu den ursprünglichen Zielen. Reflexion über Herausforderungen und Lernerfahrungen. |

## Zeitplan

## Arbeitspakete

| **ID** | **Inhalt** |
| --- | --- |
| 00 | Dokumentation der gesamten Projektarbeit, einschliesslich technischer Spezifikationen, Designentscheidungen und Implementierungsdetails. |
| 01 | Strukturierung und Planung der Dokumentationsinhalte. Hier wird festgelegt, welche Informationen wo dokumentiert werden, um eine kohärente und leicht navigierbare Dokumentation zu erstellen. Nach Wegleitung und Kriterienkatalog. |
| 02 | Überprüfung der Dokumentation auf sprachliche Korrektheit. Ziel ist es, eine professionelle und fehlerfreie Dokumentation zu gewährleisten. |
| 03 | Organisation und Durchführung von Treffen mit Expertenteam. |
| **Phase Informieren:** | |
| 04 | Analyse der Projektanforderungen und -ziele. Hier wird ein tiefes Verständnis der Aufgabenstellung entwickelt, um eine solide Basis für die Projektplanung zu schaffen. |
| 05 | Detaillierte Bewertung der Projektanforderungen anhand eines Kriterienkatalogs. Ziel ist es, Prioritäten zu setzen und sicherzustellen, dass alle Kriterien berücksichtigt werden. |
| 06 | Recherche und Sammlung von Informationen, die für die Umsetzung des Projekts relevant sind. Dies umfasst technische Daten, Branchentrends und Best Practices. |
| **Phase Planen:** | |
| 07 | Entwicklung eines detaillierten Projektzeitplans, der alle Phasen, Meilensteine und Fristen umfasst. Dieser Zeitplan dient als Wegweiser für die ganze IPA. | |
| 08 | Erstellung eines Konzepts, das den aktuellen Stand des Projekts und die Ausgangssituation beschreibt. Dies dient als Ausgangspunkt für die Entwicklung von Lösungsansätzen. | |
| 09 | Entwicklung eines umfassenden Konzepts für die Umsetzung der IPA Aufgabenstellung. | |
| 10 | Entwurf eines Datenbankmodells, das die Speicherung und Organisation aller erforderlichen Daten unterstützt. Dieses Modell wird die Grundlage für die Entwicklung der neuen Tabellen sein. | |
| 11 | Ausarbeitung eines umfassenden Testplans, der sicherstellt, dass alle Aspekte der Aufgabenstellung gründlich getestet werden, um Fehler zu minimieren und die Qualität der Endprodukte zu gewährleisten. | |
| **Phase Entscheiden:** | | |
| 12 | Entscheidung für eine spezifische Lösungsvariante nach Bewertung aller möglichen Optionen. Diese Entscheidung basiert auf einer Analyse von Vor- und Nachteilen jeder Variante im Hinblick auf die Aufgabenstellung. | |
| **Phase Realisieren:** | |
| 13 | Implementierung der Datenbank gemäss dem zuvor entworfenen Modell. Dies umfasst die Einrichtung von Tabellen und deren Beziehungen. |
| 14 | Entwicklung von Mechanismen zur Erkennung und Handhabung von Fehlern während des Datenimportprozesses vom LMS entstehen. Ziel ist es, die Integrität der in die Datenbank eingespeisten Daten sicherzustellen. |
| 15 | Entwicklung von Backend-Funktionen zur Verwaltung von Bildern, einschliesslich Upload, skalierung, Speicherung, Bearbeitung und Löschung von Bildern gemäss der Aufgabenstellung. |
| 16 | Implementierung von Sicherheitsmechanismen für das Backend, die es Benutzern ermöglichen, sich sicher einzuloggen und auszuloggen, sowie die Verwaltung von Sessions. |
| 17 | Entwicklung von Backend-Funktionen zur Verwaltung von Benutzerkonten, einschliesslich Erstellung, Bearbeitung und Deaktivierung von Benutzern. |
| 18 | Zeitraum für letzte Anpassungen am Backend und Puffer für unvorhergesehene Aufgaben oder Verzögerungen. Ziel ist es, eine stabile und vollständig funktionierende Backend-Infrastruktur sicherzustellen. |
| 19 | Anpassung und Optimierung der Kommunikation zwischen Frontend und Backend, um eine reibungslose Datenübertragung und Interaktion zu gewährleisten. |
| 20 | Implementierung von Sicherheitsmechanismen im Frontend, um den Zugriff auf geschützte Bereiche der Anwendung zu kontrollieren, sowie die Einrichtung von Routing-Regeln für eine intuitive Navigation. |
| 21 | Entwicklung einer robusten Authentifizierungsinfrastruktur im Frontend, die es Benutzern ermöglicht, sich sicher anzumelden und Zugriff auf autorisierte Inhalte zu erhalten. |
| 22 | Erstellung von Frontend-Komponenten für Login- und Logout-Funktionalitäten, einschliesslich Benutzeroberflächen für die Eingabe von Anmeldedaten und die Verwaltung von Benutzersitzungen. |
| 23 | Implementierung einer Benutzeroberfläche und Logik für den Upload von Bildern durch den Benutzer, einschliesslich Feedbackmechanismen zur Anzeige des Upload-Status. |
| 24 | Entwicklung von Frontend-Komponenten zur Anzeige von Bildern zum jeweiligen Event. |
| 25 | Erstellung von Benutzeroberflächen und Logiken im Frontend zur Verwaltung von Bildern, einschliesslich Funktionen zum Erstellen, Bearbeiten und Löschen von Bildern. |
| 26 | Entwicklung einer Benutzeroberfläche zur Anzeige von Datenimport-Transformationen, die Administratoren Einblick in Probleme gibt, die während des Imports passiert sind. |
| 27 | Implementierung einer Komponente im Frontend zur Anzeige der letzten Datenimporte, einschliesslich Status und Zeitpunkt, um Administratoren einen Überblick über Importaktivitäten zu geben. |
| 28 | Entwicklung von Benutzeroberflächen und Funktionalitäten im Frontend zur Verwaltung von Benutzerkonten, einschliesslich der Erstellung, Bearbeitung und Deaktivierung von Benutzern. |
| 29 | Zeitraum für letzte Anpassungen im Frontend und Puffer für unvorhergesehene Aufgaben oder Probleme. Ziel ist es, ein benutzerfreundliches und fehlerfreies Frontend zu gewährleisten. |
| 30 | Das Verfahren umfasst die Erstellung eines Docker-Images der Anwendung, welches anschliessend auf einem internen Linux-Server bereitgestellt wird. Zusätzlich wird eine Aktualisierung der Datenbank auf dem Server durchgeführt. |
| **Phase Kontrollieren:** | |
| 31 | Entwicklung von Testfällen basierend auf den Projektanforderungen, um eine umfassende Überprüfung der Funktionalität und Leistung der Anwendung zu ermöglichen. |
| 32 | Durchführung der erstellten Testfälle, um die korrekte Funktionsweise der Anwendung sicherzustellen und potenzielle Fehler oder Probleme zu identifizieren. |
| **Phase Auswerten:** | |
| 33 | Bewertung der implementierten Benutzerverwaltungsfunktionen, um sicherzustellen, dass sie der Aufgabenstellung entsprechen und effektiv funktionieren. | |
| 34 | Überprüfung der Bilderverwaltungsfunktionen auf ihre Effektivität und Übereinstimmung mit der Aufgabenstellung, einschliesslich der Benutzerfreundlichkeit und der Qualität der Bildbearbeitung. | |
| 35 | Analyse der Fehlerbehandlungsmechanismen, um sicherzustellen, dass Fehler effektiv erkannt und verarbeitet werden, insbesondere im Kontext des Datenimports. | |
| 36 | Rückblick auf den gesamten IPA-verlauf, um Lernerfahrungen, Herausforderungen und Erfolge zu identifizieren. Dies dient der persönlichen Entwicklung und der Verbesserung zukünftiger Projekte. | |
| 37 | Zusammenfassung der Projektergebnisse, Erkenntnisse und des erreichten Fortschritts im Vergleich zu den ursprünglichen Zielen. Bietet einen abschliessenden Überblick über die IPA. | |

## Meilensteine

| **Meilenstein** | **Geplanter Zeitpunkt** | **Bemerkungen** |
| --- | --- | --- |
| 12 | Mo 04.03.2024  17 Uhr | Verweis Arbeitspaket 12 |
| 18 | Mi 06.03.2024  15 Uhr | Verweis Arbeitspaket 18 |
| 29 | Di 12.03.2024  15 Uhr | Verweis Arbeitspaket 29 |

## Organisation der Arbeitsergebnisse

Um die Organisation der Arbeitsergebnisse umfassend zu gewährleisten, setze ich auf eine diversifizierte Strategie, die lokale Speicherung, Cloud-Dienste Google Drive für Dokumentation, sowie mehrere GitHub-Repositories einschliesst. Speziell für die Dokumentation dient das Repository "2David4/IPA-Dokumentation" (Versionierung Dokumentation) als zentrale Sammelstelle, ergänzt durch zwei weitere Repositories für den Quellcode, um eine klare Trennung und effiziente Verwaltung zu sichern.

### Versionierung

Für die tägliche Arbeit und Versionskontrolle werden spezifische Branches erstellt, die den Fortschritt dokumentieren und eine klare Struktur bieten.

### Backup

Ein automatisiertes Backup-System sichert alle Änderungen bei jedem Push auf den Master-Branch des Dokumentations-Repositories auf einen dafür abgesehen Google Drive Ordner.

### Quellcode / Skripts

Für den Sourcecode der Projekte "Competec/ita-talent-api" und "Competec /ita-talent-client" werden dedizierte Branches (Versionierung Praktische Arbeit) für die IPA erstellt, um Entwicklungen klar zu trennen und zu versionieren.

### Arbeitsplatz

Der physische Arbeitsplatz ist systematisch organisiert, wobei relevante Kriterien und Zusammenhänge visuell auf einem Bulletin Board dargestellt werden. Diese Organisation fördert nicht nur die Effizienz, sondern auch die Einhaltung der Projektanforderungen.

## Arbeitsjournal

### Tag 1 – 29.02.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
| 01 | Dokumentationsstruktur und Teil 1 der Dokumentation erstellen | Erledigt |
| 04 & 05 | Kriterienkatalog und Aufgabenstellung analysieren | Erledigt |
| 07 | Zeitplan für die kompletten 10 Tage erstellen | Erledigt |
| 03 | Expertengespräch mit dem ganzen Expertenteam auf Zoom | Erledigt |
| Ausgeführte Arbeiten: | | |
| Erstellung des Aufbaus für die Dokumentation, inklusive des Titelblatts und der Gliederung des ersten Teils des Berichts. | | |
| Ausarbeitung eines detaillierten Zeitplans für die gesamte Dauer der IPA, unterteilt in 1, 2, 4-Stunden-Blöcke, entsprechend den Empfehlungen im Kriterienkatalog. | | |
| Herausforderungen: | | |
| Beim Erstellen des Zeitplans hatte ich anfangs Schwierigkeiten, einen gut überschaubaren Plan in Excel zu erstellen, da es echt eine Herausforderung ist so viele verschiedene wichtige Daten anschaulich darzustellen. | | |
| Beanspruchte Hilfestellungen: | | |
| Keine externen Hilfestellungen wurden in Anspruch genommen. | | |
| Vergleich mit Zeitplan: | | |
| Zeitplan wurde eingehalten. | | |
| Reflexion: | | |
| Der erste Tag startete mit einer gründlichen Einarbeitung in die Projektvorgaben, was mir einen klaren Fahrplan lieferte. Die Erstellung des Zeitplans in Excel erwies sich zunächst als Herausforderung, doch durch gezieltes Spacing und Farbgebung fand ich eine effektive Lösung. Diese Anpassung steigerte nicht nur die Übersichtlichkeit, sondern auch meine Zuversicht in die kommenden Aufgaben. | | |

### Tag 2 – 04.03.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
| 06 | Recherche und Sammlung von Informationen | Erledigt |
| 08, 09, 10 | Erstellung Konzepte der Ausgangslage, Datenbank, Realisierung und eines Testplans | Erledigt |
| 12 | Entscheidung für eine spezifische Lösungsvariante (Meilenstein) | Erledigt |
| 13 | Implementierung der Datenbank | Erledigt |
| Ausgeführte Arbeiten: | | |
| Durchführung umfangreicher Recherchen zu den Themen Fehlerbehandlung, Bilderverwaltung und Benutzerverwaltung. | | |
| Implementierung der Datenbankstruktur, einschliesslich der Erstellung neuer Tabellen und Anpassung vorhandener Schemata. | | |
| Herausforderungen: | | |
| Identifizierung der optimalen Strategie zur Fehlerbehandlung und Entscheidungsfindung zwischen benutzerdefinierten Exceptions und Transformations-DTOs. | | |
| Überlegungen zur optimalen Gestaltung der neuen Tabellen und Session-Handhabung, insbesondere im Hinblick auf Sicherheit und Benutzerfreundlichkeit. | | |
| Beanspruchte Hilfestellungen: | | |
| Online-Ressourcen: <https://www.baeldung.com/java-resize-image>  <https://docs.spring.io/spring-framework/reference/data-access/oxm.html>  https://www.baeldung.com/java-reflection-class-fields | | |
| Vergleich mit Zeitplan: | | |
| Die gesetzten Ziele wurden termingerecht erreicht, was den Zeitplan bestätigt. | | |
| Reflexion: | | |
| Heute war ein intensiver, aber enorm wichtiger Tag für das Projekt. Gegen Ende liess meine Konzentration nach, doch durch das Öffnen der Fenster und das Einlassen frischer Luft fand ich neuen Schwung. Diese kleine Pause erlaubte es mir, die Energie für die Bewältigung der letzten Aufgabe des Tages - das Erstellen der Tabellen - zu sammeln. Es war befriedigend, diesen kritischen Punkt erfolgreich abzuschliessen und bestärkt das Vertrauen in meine Fähigkeit, auch zukünftige Herausforderungen zu meistern. | | |

### Tag 3 – 05.03.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
| 14 | Entwicklung von Mechanismen zur Erkennung und Handhabung von Fehlern während des Datenimportprozesses | Erledigt |
| 15 | Entwicklung von Backend-Funktionen zur Verwaltung von Bildern, einschliesslich Upload, Skalierung, Speicherung, Bearbeitung und Löschung von Bildern | Erledigt |
| Ausgeführte Arbeiten: | | |
| Ich habe mich auf die Identifizierung und Handhabung von Problemen konzentriert, die während der Verarbeitung von XML-Daten auftreten können. Für die Entwicklung von Backend-Funktionen zur Verwaltung von Bildern habe ich verschiedene Techniken zur Bildbearbeitung erforscht und implementiert, darunter das Hochladen, Skalieren, Speichern, Bearbeiten und Löschen von Bildern. | | |
| Herausforderungen: | | |
| Die grösste Herausforderung des Tages war die Behandlung von Exceptions durch XMLAdapter, da sie nicht wie erwartet funktionierten. | | |
| Beanspruchte Hilfestellungen: | | |
| <https://stackoverflow.com/questions/11114428/jaxb-xmladapter-method-does-not-throws-exception> , ChatGPT mit dieser Anfrage | | |
| Vergleich mit Zeitplan: | | |
| Trotz einiger Herausforderungen konnte ich die gesetzten Ziele für den Tag erreichen und blieb damit im Zeitplan. | | |
| Reflexion: | | |
| Heute stand ich vor einer der grössten Herausforderungen des Projekts – der Umgang mit den Eigenheiten der XMLAdapter und deren Fehlerbehandlung. Die Erkenntnis, dass diese Adapter Exceptions nicht auf herkömmliche Weise werfen, erforderte eine tiefgreifende Recherche und das Sammeln von Fachwissen. Die erfolgreiche Bewältigung dieser Schwierigkeit unterstreicht die Bedeutung von Ausdauer und Anpassungsfähigkeit in der Softwareentwicklung. Es war ein lehrreicher Tag, der mein Verständnis für spezialisierte Fehlerbehandlungsmechanismen in Java erweiterte. | | |

### Tag 4 – 06.03.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
| 16 | Implementierung Authentication (Login & Logout) | Erledigt |
| 17 | Implementierung Benutzerverwaltung | Erledigt |
| 18 | Feinschliff Backend (Meilenstein) | Erledigt |
| Ausgeführte Arbeiten: | | |
| Vollständige Implementierung der Benutzerverwaltung, einschliesslich CRUD-Operationen für Benutzer.Durchführung des Feinschliffs am Backend, vor allem das Erstellen der Endpunkte für die Transformationendetails und Details zur Bilderverwaltungs-Seite. | | |
| Herausforderungen: | | |
| Eine grosse Herausforderung war es die Gestaltung des Gesicherten Bereichs im Websockets-Listener zu implementieren.Die Implementierung der neuen Endpunkte für die detaillierte Transformations- und Event-bildanzeige verlangte eine genaue Auseinandersetzung mit den Datenstrukturen und dem Datenfluss innerhalb der Anwendung. | | |
| Beanspruchte Hilfestellungen: | | |
| Für die Implementierung der Spring Security habe ich mich an das Schema eines anderen Projekts in unserer Abteilung orientiert. <https://github.com/Competec/ita-community-api> | | |
| Vergleich mit Zeitplan: | | |
| Ich habe den Zeitplan grösstenteils eingehalten, benötigte jedoch eine zusätzliche Stunde für den Feinschliff im Backend. | | |
| Reflexion: | | |
| Heute war ein entscheidender Tag in der Entwicklung des Projekts. Ich stand vor wichtigen Entscheidungen bezüglich des Aufbaus einiger Endpunkte, die nicht nur unmittelbare Auswirkungen auf die Backend-Struktur hatten, sondern auch später entscheidend dafür sein werden, wie das Frontend gestaltet wird. Diese strategischen Entscheidungen erforderten ein tiefes Verständnis der Anforderungen und eine sorgfältige Abwägung der verschiedenen Implementierungsmöglichkeiten. | | |

### Tag 5 – 07.03.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
| 19 | Anpassung und Optimierung der Kommunikation zwischen Frontend und Backend. |  |
| 20 | Implementierung von Sicherheitsmechanismen im Frontend, um den Zugriff auf geschützte Bereiche der Anwendung zu kontrollieren. |  |
| 21 | Entwicklung einer robusten Authentifizierungsinfrastruktur im Frontend. |  |
| 22 | Erstellung von Frontend-Komponenten für Login- und Logout-Funktionalitäten. |  |
| Ausgeführte Arbeiten: | | |
|  | | |
|  | | |
| Herausforderungen: | | |
|  | | |
|  | | |
| Beanspruchte Hilfestellungen: | | |
|  | | |
| Vergleich mit Zeitplan: | | |
|  | | |
| Reflexion: | | |
|  | | |

### Tag 6 – 11.03.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Ausgeführte Arbeiten: | | |
|  | | |
|  | | |
| Herausforderungen: | | |
|  | | |
|  | | |
| Beanspruchte Hilfestellungen: | | |
|  | | |
| Vergleich mit Zeitplan: | | |
|  | | |
| Reflexion: | | |
|  | | |

### Tag 7 – 12.03.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Ausgeführte Arbeiten: | | |
|  | | |
|  | | |
| Herausforderungen: | | |
|  | | |
|  | | |
| Beanspruchte Hilfestellungen: | | |
|  | | |
| Vergleich mit Zeitplan: | | |
|  | | |
| Reflexion: | | |
|  | | |

### Tag 8 – 13.03.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Ausgeführte Arbeiten: | | |
|  | | |
|  | | |
| Herausforderungen: | | |
|  | | |
|  | | |
| Beanspruchte Hilfestellungen: | | |
|  | | |
| Vergleich mit Zeitplan: | | |
|  | | |
| Reflexion: | | |
|  | | |

### Tag 9 – 14.03.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Ausgeführte Arbeiten: | | |
|  | | |
|  | | |
| Herausforderungen: | | |
|  | | |
|  | | |
| Beanspruchte Hilfestellungen: | | |
|  | | |
| Vergleich mit Zeitplan: | | |
|  | | |
| Reflexion: | | |
|  | | |

### Tag 10 – 18.03.2024

| ID | Tagesziele | Status |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Ausgeführte Arbeiten: | | |
|  | | |
|  | | |
| Herausforderungen: | | |
|  | | |
|  | | |
| Beanspruchte Hilfestellungen: | | |
|  | | |
| Vergleich mit Zeitplan: | | |
|  | | |
| Reflexion: | | |
|  | | |

Teil 2 – Projekt

## Kurzfassung

### Ausgangslage

…

### Umsetzung

…

### Ergebnis

## Phase "Informieren"

### Strategien zur Fehlerbehandlung

Bei der Weiterentwicklung des "CompAcademy Lernportals" habe ich mich intensiv mit dem Teilauftrag 3 auseinandergesetzt, der sich auf die Fehlerbehandlung, den Einsatz von XML-Adaptern und die Entwicklung von einer Fehlerbehandlung konzentriert. Während dieser Informationsphase habe ich zwei Hauptansätze zur Fehlerbehandlung evaluiert und zudem eine Methode zur Identifizierung von Problemen in generischen XML-Adaptern recherchiert.

Fehlerbehandlung: Zwei Hauptansätze

**1.** **Benutzerdefinierte Exceptions:** Diese Variante sieht vor, dass der XML-Adapter bei Auftreten eines Fehlers eine spezifisch definierte Exception wirft. Dieser Ansatz ermöglicht eine direkte und klare Kommunikation von Fehlern, stellt aber die Herausforderung dar, den Fehlerkontext präzise zu übermitteln.

**2. Transformations-DTO mit Fehlerdetails:** Im Gegensatz zum direkten Werfen von Exceptions sieht dieser Ansatz vor, dass der XML-Adapter ein Transformations-DTO zurückgibt, das während der Verarbeitung erstellt wird. Fehlerdetails werden in diesem DTO festgehalten, was eine umfassende Sicht auf den Fehlerkontext ermöglicht, ohne den Prozessfluss durch Exceptions zu unterbrechen.

**Dynamische Feldidentifikation**

Ein signifikantes Problem in beiden Ansätzen ist die Identifikation des spezifischen Feldes, das den Fehler verursacht, insbesondere weil die XML-Adapter generisch konzipiert sind und für mehrere Felder verwendet werden. In den XML-Adaptern selbst sind weder die ID noch der Feldname vorhanden, lediglich der Wert, was die Zuweisung und Rückverfolgung von Fehlern erschwert.

Um dieses Problem zu lösen, habe ich mich mit Techniken zur dynamischen Feldidentifikation in Java beschäftigt. Der Schlüssel liegt in der Verwendung der Java Reflection, speziell der Methode getDeclaredFields(), die es ermöglicht, alle Felder eines DTOs zur Laufzeit abzurufen. Diese Information kann dann genutzt werden, um in einer ausführlichen switch-Case-Logik spezifische Behandlungen oder Fehlermeldungen für jedes Feld zu definieren.

Diese Methode erlaubt es uns, flexible und wiederverwendbare XML-Adapter zu implementieren, die nicht nur auf die aktuelle Event-Datenstruktur beschränkt sind, sondern auch für zukünftige Anwendungen und Datenimporte, wie beispielsweise aus MyAbacus, adaptiert werden können.

**Quellen und Recherche**

Die Entwicklung dieser Ansätze basierte auf einer Kombination aus Fachliteratur, Online-Ressourcen und Diskussionen in Entwicklergemeinschaften. Insbesondere Artikel über Java Reflection und benutzerdefinierte Exceptions boten tiefgreifende Einblicke und praxisnahe Lösungsansätze.

Informationen zur Feldidentifikation: <https://www.baeldung.com/java-reflection-class-fields>

Informationen zum Marshalling Prozess der XML-Adapter: <https://docs.spring.io/spring-framework/reference/data-access/oxm.html>

Informationen zu Exception Bubbling und Best Practices: <https://coding-examples.com/java/exception-bubbling-in-java/>

**Zusammenfassung**

Die sorgfältige Erkundung und Bewertung der Fehlerbehandlungsoptionen sowie der dynamischen Feldidentifikation ohne den Einsatz von Spring bildeten eine solide Grundlage für die Implementierung effektiver Strategien im "CompAcademy Lernportal". Diese Erkenntnisse tragen wesentlich zur Verbesserung der Datenverarbeitungsrobustheit bei und ermöglichen eine flexible Erweiterung der Plattform für zukünftige Anforderungen.



### Informationsbeschaffung Bilderverwaltung

Diese Phase der Informationsbeschaffung vom Teilauftrag 2 konzentrierte sich auf die Integration von Spring Security, die Handhabung von Bildern mittels MultipartFile, Bildskalierung und -validierung. Ziel war es, eine sichere und benutzerfreundliche Lösung zu schaffen, die den visuellen Ansprüchen des Lernportals gerecht wird und dabei die technischen Anforderungen der Aufgabenstellung erfüllt.

**Sicherheitskonzepte mit Spring Security**

Die Einführung von Spring Security dient der Absicherung der Bildverwaltungsfunktionen, um sicherzustellen, dass nur autorisierte Benutzer Zugriff auf das Hochladen und Verwalten von Bildern haben. Dies erforderte eine tiefgehende Recherche von der Spring Security Dokumentation zu Authentifizierungs- und Autorisierungsmechanismen in Spring-basierten Anwendungen.

**Effiziente Bildverarbeitung**

Ein zentraler Aspekt der Bildverwaltung ist die effektive Handhabung von Bilduploads. Hierbei erwies sich MultipartFile als Schlüsselkomponente für das Empfangen und Verarbeiten von Bilddateien. Die Untersuchung der Spring File Upload Dokumentation galt der optimalen Anwendung von MultipartFile für den Bildupload-Prozess.

**Bildskalierung und Formatvalidierung**

Das Skalieren von Bildern, um eine maximale Breite von 1024px zu gewährleisten, und die Validierung der Bildformate (ausschliesslich PNG und JPG) waren weitere kritische Themen. Informationen zur Implementierung von Bildskalierungsmethoden wie Java Bildskalierung Dokumentation und Validierungslogiken waren von entscheidender Bedeutung, um die Konsistenz und Qualität der Bildinhalte zu sichern.

**Varianten der Endpunktgestaltung**

In der Informationsphase stiess ich auf zwei Varianten zur Gestaltung der Endpunkte für den Bildupload:

**Variante 1:** Nutzt @ModelAttribute zur automatischen Zuordnung der eingehenden Daten zu einem DTO.

**Variante 2:** Verwendet @RequestParam für eine explizite Definition und Verarbeitung der Upload-Parameter.

Beide Ansätze bieten spezifische Vorteile hinsichtlich Flexibilität und Strukturierung der Datenübermittlung, was eine fundierte Entscheidungsgrundlage für die Implementierung bildet.

**Quellen und Recherche**

Die Informationsphase stützte sich auf eine Vielzahl von Ressourcen, darunter:

Spring Security Dokumentation: <https://docs.spring.io/spring-security/reference/servlet/configuration/java.html>

Spring File Upload Dokumentation: <https://www.baeldung.com/spring-boot-thymeleaf-image-upload>

Java Bildskalierung Dokumentation: <https://www.baeldung.com/java-resize-image>

Diese Quellen boten wertvolle Einblicke in bewährte Methoden und technische Lösungen.

**Zusammenfassung**

Die gründliche Informationsbeschaffung zu den Kernaspekten der Bildverwaltung hat eine solide Basis für die Entwicklung der Bildverwaltungsfunktionen im "CompAcademy Lernportal" geschaffen. Durch die Bewertung von Sicherheitsaspekten, Bildverarbeitungstechniken und Endpunktgestaltungsvarianten ist es nun möglich, eine Lösung zu implementieren, die sowohl den technischen Anforderungen als auch den Nutzererwartungen entspricht. Die nächste Phase wird die praktische Umsetzung der gesammelten Erkenntnisse umfassen, mit dem Ziel, eine optimierte und sichere Bildverwaltung im Lernportal zu realisieren. Somit wäre das Arbeitspaket 06 erledigt



## Phase "Planen"

### Ist-Zustand des Systems

Im "Ist-Zustand des Systems" der IPA zur Erweiterung des "CompAcademy Lernportals" zeigt die initiale Skizze deutlich die bestehende Systemarchitektur und die Kommunikationspfade zwischen den Schlüsselkomponenten. Ein farbcodiertes Schema hilft dabei, die unterschiedlichen Kommunikationsmethoden im System zu veranschaulichen: Blau kennzeichnet die Verwendung von REST-APIs, während Grün für die bereits etablierte Websocket-Kommunikation steht.

Die Systemstruktur besteht aus mehreren Hauptkomponenten:

* Client (UI): Entwickelt mit React, stellt diese Komponente eine dynamische Schnittstelle für die Nutzer dar. Durch die Websocket-Kommunikation (grün dargestellt) wird eine direkte und reaktive Verbindung zum Backend hergestellt, was die Aktualität der Daten auf der Benutzeroberfläche gewährleistet.
* Backend (API): Das Backend, strukturiert um Websockets, nutzt Listener und Responder zusammen mit einem Exchange DTO, um eine effiziente und reaktive Datenverarbeitung sowie -übertragung zu ermöglichen. Diese Architektur unterstützt eine nahtlose Echtzeit-Kommunikation zwischen Client und Server.
* Datenbank (DB): In der PostgreSQL-Datenbank werden kritische Daten wie Kursinformationen gespeichert. Die enge Integration mit dem Backend ermöglicht eine effiziente Verwaltung und Abfrage dieser Daten.
* LMS: Das externe Learning Management System (LMS) von Umantis AG versorgt das System regelmässig mit den notwendigen Kursdaten. Über einen REST-Endpunkt (in Blau hervorgehoben) werden diese Informationen alle 10 Minuten abgerufen, um sie anschliessend im Lernportal zu konvertieren, zu normalisieren und in der Datenbank zu speichern.

Die Skizze des "Ist-Zustands des Systems" unterstreicht die zentrale Rolle der Websocket-basierten Backend-Architektur, die eine effektive Verbindung zwischen dem Client, der Datenbank und dem LMS herstellt. Die Verwendung von Farbcodes erleichtert das Verständnis der verschiedenen Kommunikationswege und liefert eine klare Basis für das Gesamtbild der Systemarchitektur. Damit wäre das Arbeitspaket 08 abgehackt.

### Soll-Zustand des Systems

Im "Soll-Zustand des Systems" für der IPA zur Erweiterung des "CompAcademy Lernportals" zeigt die aktualisierte Skizze die geplanten Neuerungen und Erweiterungen, die während dieser IPA implementiert werden sollen wie im Arbeitspaket 09 verlangt. Die Skizze nutzt ein farbcodiertes Schema, in dem die neu hinzukommenden Features in Rot hervorgehoben sind, um einen klaren Überblick über die geplanten Änderungen zu geben. Diese Neuerungen umfassen:

* Benutzerverwaltung (Person): Eine neue Komponente zur Verwaltung der Benutzerdaten, die es ermöglicht, Benutzerprofile effizient zu erstellen, zu bearbeiten und zu verwalten.
* Authentikation und Sessionhandling (Session): Diese Funktionalität soll ein sicheres Login-Verfahren und eine effiziente Verwaltung von Benutzersitzungen gewährleisten. Sie ist essenziell für die Sicherheit und Personalisierung des Nutzererlebnisses.
* Anzeige von Fehlern (Transformation): Eine wichtige Erweiterung zur Visualisierung von Fehlern, die während der Datenkonvertierung und -normalisierung auftreten können. Dies ermöglicht eine schnellere Identifizierung und Behebung von Problemen.
* Imports (Datum des Imports und Status): Die Implementierung einer Übersicht über die Importvorgänge, einschliesslich Zeitpunkt und Status, bietet Transparenz und Kontrolle über den Datenimport aus dem LMS.
* Bilderverwaltung (Image): Um die Kursangebote visuell ansprechender zu gestalten, wird eine vollständige Bilderverwaltung eingeführt. Diese Komponente wird über REST-Endpunkte angesprochen, um eine effiziente Verarbeitung und Darstellung von Bildern zu ermöglichen, getrennt von der Textlastigen Kommunikation via Websockets.

Die Entscheidung, für die Bilderverwaltung auf REST-Endpunkte zu setzen, gründet sich nicht nur auf die technische Notwendigkeit, die Ladezeiten für Bilder zu optimieren und eine unabhängige Handhabung von Bild- und Textinhalten zu ermöglichen, sondern folgt auch explizit den Vorgaben der Aufgabenstellung (Bilder werden über eine REST-Schnittstelle angefragt, hochgeladen, angepasst und gelöscht.). Dieser Ansatz erlaubt es, Bilder effizient und getrennt vom textbasierten Inhalt zu laden, was einen erheblichen Beitrag zur Performancesteigerung und Verbesserung der Benutzerfreundlichkeit des Lernportals leistet. Die klare Trennung von Bild- und Textdaten durch die Nutzung von REST-Endpunkten für die Bilderverwaltung entspricht somit nicht nur den technischen Bestrebungen nach einer optimierten Nutzererfahrung, sondern erfüllt auch die spezifischen Anforderungen, die in der Aufgabenstellung für dieses Projekt festgelegt wurden.

Durch diese Erweiterungen wird das Lernportal nicht nur um wichtige Funktionen reicher, sondern auch in Bezug auf Sicherheit, Benutzerfreundlichkeit und Effizienz deutlich verbessert. Die Integration neuer Features im "Soll-Zustand des Systems" markiert einen entscheidenden Schritt vorwärts in der Entwicklung des "CompAcademy Lernportals", um den Anforderungen und Erwartungen der Nutzer gerecht zu werden und die Verwaltung von Kursangeboten innerhalb der Competec-Gruppe zu optimieren.

### Datenbankmodell



|  |  |
| --- | --- |
|  | **Bestehende Tabellen (vor der IPA)** |
|  | **Neue Tabellen (während der IPA hinzugefügt)** |

Zum Arbeitspaket 10 und zu der Planung meiner IPA zur Erweiterung des "CompAcademy Lernportals" habe ich ein detailliertes Datenbankmodell entworfen, das die Kernstrukturen und die neu einzuführenden Elemente klar darstellt. Durch ein farbcodiertes Schema werden die bereits existierenden Bestandteile (in Blau) und die im Zuge der IPA hinzugefügten Erweiterungen (in Grün) unterschieden. Dieser Ansatz gewährleistet eine sofortige visuelle Unterscheidung und erleichtert das Verständnis der Datenbankarchitektur.

**Bestehende Strukturen und Beziehungen**

Das Modell zeigt die bestehenden Tabellen wie Event, Event\_Subcategory, Event\_Speaker, Event\_Event\_Speaker und Search, die die Basis des Lernportals bilden. Die Event-Tabelle, mit einer Selbstbeziehung über das Feld parent\_id, ermöglicht die Abbildung von Event-Hierarchien. Diese Struktur stellt sicher, dass Haupt- und Unter-Events effizient verwaltet werden können.

**Neue Erweiterungen in der Datenbank**

Während meiner IPA führe ich neue Tabellen ein:

* **Person:** Für die erweiterte Benutzerverwaltung, die die Grundlage für personalisierte Nutzererfahrungen im Lernportal schafft.
* **Person\_Session:** Unterstützt das neue Session-Handling, das für die Authentifizierung und Sessionverwaltung unerlässlich ist.
* **Image:** Ermöglicht das Hinzufügen von Bildern zu Kursen, was die Attraktivität der Kursangebote steigert.
* **Event\_Transformation und Event\_Import:** Diese Tabellen dienen der Überwachung und Verwaltung von Datenimport- und Transformationsprozessen, was eine detaillierte Fehleranalyse und -behandlung ermöglicht.

**Selbstbeziehung und Datenbeziehungen**

Ein besonderes Merkmal ist die Selbstbeziehung in der Event-Tabelle, die eine flexible Darstellung von Event-Strukturen erlaubt. Diese Konzeption spiegelt die Komplexität und Dynamik der Eventverwaltung wider und bietet eine solide Basis für die Darstellung von Event-Beziehungen im Lernportal.

**Zusammenfassung**

Das entworfene Datenbankmodell bildet eine wesentliche Grundlage für die erfolgreiche Umsetzung meiner IPA. Die klare Unterscheidung zwischen bestehenden und neuen Datenstrukturen durch Farbcodierung verbessert nicht nur die Übersichtlichkeit, sondern erleichtert auch die zukünftige Entwicklung und Erweiterung der Plattform. Die Planung dieses Datenbankmodells ist ein entscheidender Schritt, um die Funktionalität und Benutzerfreundlichkeit des "CompAcademy Lernportals" zu erweitern und zu optimieren. Im weiteren Verlauf meiner IPA wird dieses Modell als Richtlinie für die Entwicklung und Implementierung der Datenbankstruktur dienen.

### Tabellenstruktur

Im Rahmen meiner IPA zur Erweiterung des "CompAcademy Lernportals" ist die Entwicklung einer robusten Datenbankstruktur von zentraler Bedeutung. Diese Struktur soll nicht nur die spezifischen Anforderungen der drei Teilaufträge erfüllen, sondern auch die Grundlage für zukünftige Erweiterungen bieten. Jede Tabelle folgt einem einheitlichen Schema für Metadaten, das die Nachverfolgbarkeit und Verwaltung der Datensätze erleichtert:

**Standardmetadaten**

* date\_created: Zeitstempel der Erstellung des Datensatzes.
* user\_created: Benutzer, der den Datensatz erstellt hat.
* date\_modified: Zeitstempel der letzten Änderung des Datensatzes.
* user\_modified: Benutzer, der den Datensatz zuletzt geändert hat.
* is\_deleted: Markierung, ob der Datensatz als gelöscht betrachtet wird.

Nachfolgend sind die spezifischen Tabellen und deren Felder beschrieben, die zur Unterstützung der geforderten Funktionalitäten implementiert werden:

**Teilauftrag 1: Login und Benutzerverwaltung**

**Tabelle: person**

* id (UUID): Eindeutiger Identifikator für jede Person.
* name (VARCHAR): Benutzername, einzigartig.
* password (VARCHAR): Verschlüsseltes Passwort.
* email (VARCHAR): E-Mail-Adresse, einzigartig und für die Identifikation beim Login verwendet.
* is\_active (BOOLEAN): Zeigt an, ob das Benutzerkonto aktiv ist. Alternativ kann is\_deleted aus den Standardmetadaten verwendet werden. (siehe Gedankengänge)
* Standardmetadaten: Wie oben beschrieben.

**Tabelle: person\_session**

* id (UUID): Eindeutiger Identifikator für jede Session. (siehe Gedankengänge)
* person\_id (UUID): Fremdschlüssel, der auf die person Tabelle verweist und den Benutzer der Session identifiziert.
* token (VARCHAR): Einzigartiger Token für die Session, der für die Authentifizierung bei API-Anfragen verwendet wird.
* expires (TIMESTAMP): Das Ablaufdatum und -zeit der Session.
* Standardmetadaten: Wie oben beschrieben.

**Gedankengänge Person Tabelle**

**Variante mit is\_active:** Ermöglicht explizite Kontrolle über den Status eines Benutzers (aktiv/inaktiv), was besonders nützlich ist, um temporäre Deaktivierungen zu verwalten.

**Variante mit is\_deleted als Ersatz für is\_active:** Vereinfacht die Struktur, indem nur ein Feld für die Statusverwaltung (gelöscht/nicht gelöscht) genutzt wird. Könnte jedoch die Flexibilität in der Verwaltung von Benutzerkonten einschränken.

**Gedankengänge Person\_session Tabelle**

**Primärschlüssel:** Die Verwendung einer UUID für id ermöglicht eine eindeutige Identifizierung jeder Session.

**Token als Primärschlüssel?:** Eine Überlegung könnte sein, den Token selbst als Primärschlüssel zu verwenden, um die Eindeutigkeit zu garantieren. Allerdings könnten sich dadurch Einschränkungen in der Flexibilität ergeben.

**Teilauftrag 2: Bilderverwaltung**

**Tabelle: image**

* event\_id (INT) / id (UUID): Als Primärschlüssel genutzt, wobei event\_id die Einzigartigkeit pro Event sicherstellt. (siehe Gedankengänge)
* image (BYTEA): Das eigentliche Bild.
* description (VARCHAR): Bildbeschreibung.
* Standardmetadaten: Wie oben beschrieben.

**Gedankengänge zur Image Tabelle**

Die Gestaltung der Image-Tabelle bedarf einer sorgfältigen Überlegung, insbesondere im Hinblick auf die Verwaltung der Beziehung zwischen Bildern und Events. Hier sind zwei Varianten und die dazugehörigen Gedankengänge:

**Variante 1:** Verwendung von event\_id als Primärschlüssel (PK)

Ein Bild pro Event: Durch die Nutzung der event\_id als Primärschlüssel wird sichergestellt, dass jedes Event genau ein Bild zugeordnet haben kann. Dies vereinfacht die Zuordnung und vermeidet Redundanzen.

**Aktualisierung und Soft-Delete:** Wenn ein Bild zu einem Event aktualisiert oder gelöscht werden soll, würde der is\_deleted-Status des vorhandenen Datensatzes verwendet:

**Bildlöschung:** Setzt is\_deleted auf true, behält aber den Datensatz in der Datenbank. Dies ermöglicht eine einfache Wiederherstellung und beibehält die Historie.

Bildaktualisierung: Beim Hinzufügen eines neuen Bildes zu einem Event, das bereits ein Bild hat, müsste zunächst geprüft werden, ob bereits ein Datensatz mit der entsprechenden event\_id existiert und is\_deleted auf false gesetzt werden. Das neue Bild ersetzt das alte, und der Datensatz wird aktualisiert, anstatt einen neuen zu erstellen.

**Variante 2:** Verwendung von UUID als Primärschlüssel

Mehr Flexibilität: Die Verwendung einer UUID als Primärschlüssel erlaubt es, mehrere Bilder pro Event zu haben, wobei zu jedem Zeitpunkt nur eines als aktuell markiert ist (is\_deleted = false). Dies könnte die Verwaltung von Bildversionen oder temporären Bildaktualisierungen erleichtern.

Vereinfachte Logik für Hinzufügen und Löschen: Mit einer UUID als Primärschlüssel wäre die Logik für das Hinzufügen und Löschen von Bildern direkter. Ein neues Bild hinzuzufügen, würde einfach einen neuen Datensatz erstellen, ohne den Status eines vorhandenen Datensatzes überprüfen zu müssen. Ebenso würde das Löschen eines Bildes einfach den is\_deleted-Status des zugehörigen Datensatzes ändern, ohne sich um die event\_id-Uniqueness sorgen zu müssen.

**Teilauftrag 3: Fehlerbehandlung des Datenimports**

**Tabelle: event\_transformation**

* id (UUID): Eindeutiger Identifikator der Transformation.
* event\_id (INT): Verweist auf die betroffene Event-ID.
* title (VARCHAR): Titel der Transformation.
* type\_enum (VARCHAR): Typ der Transformation (z.B. Kritisch, Konvertierung).
* cause (VARCHAR): Ursache der Transformation.
* effect (VARCHAR): Effekt der Transformation auf die Daten.
* Standardmetadaten: Wie oben beschrieben.

**Event\_Import**

* id (UUID): Eindeutiger Identifikator des Importvorgangs.
* status\_enum (VARCHAR): Status des Importvorgangs (z.B. OK, Fehler).
* import\_time (TIMESTAMP): Optional, für den genauen Zeitpunkt des Importbeginns, falls eine Unterscheidung von date\_created notwendig ist. (siehe Gedankengänge)
* Standardmetadaten: Wie oben beschrieben.

**Gedankengänge zu Importzeitpunkten:**

**Verwendung von date\_created:** Dieser Ansatz würde den Zeitpunkt erfassen, zu dem der Importdatensatz in der Datenbank erstellt wurde. Obwohl dies eine einfache Lösung darstellt, spiegelt es möglicherweise nicht den tatsächlichen Beginn des Importprozesses wider, vor allem, wenn zwischen dem Start des Imports und der Datensatzerstellung in der Datenbank eine Verzögerung besteht.

**Einführung eines zusätzlichen Feldes import\_time:** Diese Variante erlaubt die explizite Speicherung des genauen Zeitpunkts, zu dem der Import gestartet wurde, unabhängig vom Zeitpunkt der Datensatzerstellung (date\_created). Dies bietet eine präzisere Nachverfolgung der Importvorgänge, erhöht jedoch die Komplexität des Datenmodells und des Importprozesses, da der Importzeitpunkt explizit im Code festgehalten und beim Erstellen des Datensatzes eingefügt werden muss.

### Testkonzept

**Einleitung**

Wie im Arbeitspaket 11 festgehalten wollte ich ein Testkonzept erstellen. Ziel dieses Testkonzepts ist es, einen strukturierten Ansatz für die Validierung der Funktionalitäten und Sicherheitsaspekte des erweiterten Lernportals zu bieten. Es umfasst die systematische Durchführung von Tests zur Überprüfung der Anforderungen aus den Teilaufträgen: Login und Benutzerverwaltung, Bilderverwaltung sowie Fehlerbehandlung des Datenimports.

**Testziele**

1. Sicherstellen, dass alle Komponenten gemäss den definierten Spezifikationen der Aufgabenstellung funktionieren.
2. Identifizierung und Behebung von Sicherheitslücken innerhalb des Systems.
3. Gewährleistung einer hohen Benutzerfreundlichkeit und Zugänglichkeit der neuen Features.

**Testumgebung**

Lokale Entwicklungsumgebung mit Zugriff auf die vollständige Software- und Datenbankstruktur des Lernportals.

Verwendung von Testdaten, die repräsentativ für die Produktionsdaten sind.

**Testarten**

* Funktionstests: Überprüfen der Funktionalität und des Verhaltens der neuen Features unter verschiedenen Bedingungen.
* Sicherheitstests: Identifizierung von Sicherheitsrisiken, insbesondere im Bereich der Benutzerverwaltung und Bilderverwaltung.
* Usability-Tests: Bewertung der Benutzerfreundlichkeit und Zugänglichkeit der neuen Features für Endbenutzer.

**Testfall**

| ID / Bezeichnung | *T-001* | *gemäss Testkonzept* |
| --- | --- | --- |
| Beschreibung | *gemäss Testkonzept* | |
| Testvoraussetzung | *gemäss Testkonzept* | |
| Testschritte | *gemäss Testkonzept* | |
| Erwartetes Ergebnis | *gemäss Testkonzept* | |

**Testdurchführung und Testergebnis (Mängelklasse)**

|  |  |
| --- | --- |
| Testdatum |  |
| Tester |  |
| Mängelklasse\* |  |
| Mangelbeschreibung |  |
| Bemerkungen |  |
| *\*Mängelklasse: 0 = mängelfrei; 1 = belangloser Mangel; 2 = leichter Mangel; 3 = schwerer Mangel; 4 = kritischer Mangel* | |

## Phase "Entscheiden"

### Einleitung

Im Rahmen des Arbeitspakets 12 und der IPA der Erweiterung des CompAcademy Lernportals standen mehrere wichtige Entscheidungen an, die basierend auf gründlichen Überlegungen und Analysen getroffen wurden. Diese Entscheidungen betreffen die Fehlerbehandlung, die Bilderverwaltung, die Benutzerverwaltung, die Sessionverwaltung und die Handhabung der Importzeitpunkte.

### Fehlerbehandlung

Nach sorgfältiger Abwägung der beiden Varianten zur Fehlerbehandlung habe ich mich für die Benutzerdefinierte Exceptions entschieden. Der primäre Grund für diese Entscheidung liegt in der direkten und klaren Kommunikation von Fehlern, die es ermöglicht, Probleme effizient zu identifizieren und zu beheben. Obwohl diese Methode die Herausforderung mit sich bringt, den Fehlerkontext präzise zu übermitteln, wurde sie aufgrund ihrer Direktheit und Einfachheit als der beste Ansatz für meine IPA bewertet.

### Bilderverwaltung

Für die Implementierung der Bilderverwaltung wurde die Variante 1 mit @ModelAttribute ausgewählt. Diese Entscheidung basiert auf der Einfachheit und Klarheit der automatischen Zuordnung eingehender Daten zu einem DTO, was den Prozess des Bilduploads und der Datenverarbeitung vereinfacht. Diese Methode unterstützt eine strukturierte und übersichtliche Handhabung von Bildern im System und fördert die Entwicklungseffizienz.

### Benutzerverwaltung

In der Person-Tabelle wurde die Variante mit is\_active bevorzugt. Diese Entscheidung wurde getroffen, um eine explizite Kontrolle über den Status eines Benutzers (aktiv/inaktiv) zu ermöglichen. Diese Flexibilität ist besonders wichtig für die zukünftige Entwicklung und Erweiterbarkeit des Systems, einschliesslich der potenziellen Einführung eines Rollensystems oder der Handhabung temporärer Deaktivierungen.

### Sessionverwaltung

Bei der Gestaltung der person\_session Tabelle entschied ich mich für die Verwendung einer UUID als Primärschlüssel und behielt den Token als separates Attribut bei. Diese Entscheidung ermöglicht eine eindeutige Identifizierung jeder Session und behält gleichzeitig die Flexibilität bei, die für zukünftige Erweiterungen wie die Reaktivierung von Tokens notwendig ist. Es entspricht den Best Practices und unterstützt eine robuste Sessionverwaltung.

### Image Tabelle

Für die Image-Tabelle habe ich mich für die Variante 1 entschieden, die event\_id als Primärschlüssel zu verwenden. Diese Entscheidung stellt sicher, dass jedes Event maximal ein Bild haben kann, was die Datenintegrität und -verwaltung vereinfacht. Obwohl dies die Komplexität bei der Bildaktualisierung und beim Soft-Delete erhöht, wurde es als wertvolle Investition für ein sauberes und gut verwaltetes System erachtet.

### Importzeitpunkte

Hinsichtlich der Handhabung der Importzeitpunkte entschied ich mich gegen die Einführung eines neuen Feldes für import\_time und für die Verwendung von date\_created. Der Mehrwert einer Millisekunden genauen Erfassung des Importzeitpunkts rechtfertigt nicht die erhöhte Komplexität und den zusätzlichen Entwicklungsaufwand. Stattdessen wird das bestehende Feld date\_created verwendet, um den Zeitpunkt der Datensatzerstellung zu erfassen, was für die Zwecke der Administratoren ausreichend genau ist.

### Fazit

Mit dem Abschluss der Phasen Informieren, Planen und Entscheiden habe ich einen signifikanten Meilenstein 12 in meinem Projekt zur Erweiterung des Lernportals erreicht. Diese Anfangsphasen waren entscheidend, um eine solide Grundlage für die kommenden Schritte zu legen und sicherzustellen, dass das Projekt auf einem durchdachten und gut strukturierten Konzept aufbaut.

Jetzt, wo das Stadium des Informierens abgeschlossen ist und die Planungs- sowie Entscheidungsphasen erfolgreich durchlaufen wurden, fühle ich mich bereit und motiviert, mit der Realisierung des Projekts zu beginnen. Die Vorfreude, die integrierte Entwicklungsumgebung (IDE) zu starten und die ersten Zeilen Code zu schreiben, ist gross. Es ist spannend, die geplanten Features und Funktionalitäten zum Leben zu erwecken und zu sehen, wie das Lernportal Schritt für Schritt Gestalt annimmt.

Ich bin zuversichtlich, dass die bislang geleistete Vorarbeit eine effiziente und zielgerichtete Umsetzung ermöglicht. Die Entscheidungen, die getroffen wurden, geben mir die Sicherheit, dass die gewählten Lösungen den Anforderungen entsprechen und zur erfolgreichen Erweiterung des Lernportals beitragen werden.

Darüber hinaus bestätigt der aktuelle Stand, dass ich gut im Zeitplan liege. Diese Tatsache stärkt mein Vertrauen in die weitere Projektarbeit und motiviert mich zusätzlich, die anstehenden Aufgaben mit Engagement und Sorgfalt anzugehen. Die kommende Phase der Realisierung wird zweifellos herausfordernd sein, doch ich bin bereit, diese Herausforderungen anzunehmen und das Projekt erfolgreich zum Abschluss zu bringen.

## Phase "Realisieren"

### Realisierungskonzept

Diese Reihenfolge basiert auf der Einschätzung der Komplexität, der Notwendigkeit von Forschungsarbeit sowie der Erfahrung in den jeweiligen Bereichen.

Der Anfang macht das Arbeitspaket 13. Bevor mit der spezifischen Fehlerbehandlung und Bildverwaltung begonnen wird, ist es notwendig, eine solide Datenbankstruktur zu erstellen, die alle erforderlichen Daten effizient speichert und verarbeitet.

Danach kommt das Arbeitspaket 14 Fehlerbehandlung des Datenimports backend, da dieses Gebiet als besonders herausfordernd gilt. Die Spezifität des Projekts bedeutet, dass wenig Vorwissen oder Forschungsmaterial verfügbar ist, was dieses Paket zu einem idealen Startpunkt macht, um frühzeitig auf mögliche Schwierigkeiten zu stossen und Lösungen zu entwickeln.

Anschliessend wird das Arbeitspaket 15 Implementierung der Bildverwaltung backend in Angriff genommen. Obwohl auch hier weniger Routine vorhanden ist, bietet die Thematik im Vergleich zum vorherigen Paket mehr Forschungsmöglichkeiten, was die Implementierung erleichtern sollte.

Mit diesen Grundlagen wird der Fokus dann auf die Arbeitspakete 16 Autorisierung (Login & Logout) backend und 17 Implementierung Benutzerverwaltung backend gelegt. Diese bilden das Rückgrat der Benutzerverwaltung und -sicherheit im System und sind entscheidend für die Funktionstüchtigkeit des Lernportals.

Nachdem eine solide Backend-Struktur etabliert wurde, beginnt die Arbeit am Frontend, beginnend mit dem Arbeitspaket 19 Anpassung Kommunikation frontend, gefolgt von 20 Zugriffsschutz und Routing frontend und 21 Authentifizierungsarchitektur frontend. Diese Schritte stellen sicher, dass die Basis für eine sichere und intuitive Benutzeroberfläche gelegt ist.

Die Implementierung der Benutzerinteraktionen im Frontend, wie 22 Login-Logout-Komponente frontend, 23 Bildupload frontend, 24 Visuelle Darstellung der Bilder frontend, und 25 Implementierung Bilderverwaltung frontend, erfolgt im Anschluss. Diese Komponenten sind essenziell für die Benutzererfahrung und ermöglichen es den Nutzern, mit dem Lernportal auf produktive Weise zu interagieren.

Zum Abschluss des Frontend-Teils wird mit den Arbeitspaketen 26 Anzeige der Transformationen frontend und 27 Anzeige der Imports frontend eine direkte Rückmeldung über die Datenverarbeitungsprozesse im Backend an die Administratoren des Lernportals ermöglicht. Die abschliessende Arbeit am Frontend wird mit dem Arbeitspaket 28 Implementierung der Benutzerverwaltung frontend vollendet, was den Kreis zur Backend-Arbeit schliesst und eine kohärente Benutzerverwaltung sowohl im Backend als auch im Frontend sicherstellt.

### Arbeitspaket 13

Im Rahmen der Erweiterung des Lernportals und insbesondere mit der Einführung eines Benutzermanagements war es notwendig, diese Metadaten um user\_created und user\_modified zu erweitern. Diese Anpassung trägt der neuen Funktionalität Rechnung, dass Benutzer nun direkt Inhalte bearbeiten können, und ermöglicht eine präzisere Nachverfolgung von Änderungen an den Daten durch spezifische Benutzerkonten.

**Aktualisierung der Infrastruktur**

Entsprechend habe ich beschlossen, die gesamte Datenbankinfrastruktur mit diesen zusätzlichen Metadaten zu aktualisieren. Dieser Schritt sorgt für eine durchgängige Konsistenz in der Datenverwaltung und ermöglicht es, Änderungen an den Daten nicht nur zeitlich, sondern auch hinsichtlich der verantwortlichen Benutzer zu erfassen. Die Erweiterung der Metadatenfelder um user\_created und user\_modified ist ein wichtiger Baustein, um die Transparenz und Nachvollziehbarkeit innerhalb des Lernportals zu erhöhen.

**Umsetzung der Tabellenstruktur**

Bei der Implementierung der neuen Tabellen event\_transformation, event\_import, person, person\_session, und image habe ich mich an den Entscheidungen der Phase "Entscheiden" orientiert, die ich zuvor im Rahmen des Entscheidungsprozesses getroffen habe. Dazu gehören die Nutzung von event\_id als Primärschlüssel in der image-Tabelle und die Einführung von user\_created und user\_modified als Teil der Metadaten in allen Tabellen. Diese Entscheidungen spiegeln die Anforderungen der Teilaufträge wider und berücksichtigen sowohl technische Notwendigkeiten als auch die Erwartungen der Endbenutzer.

**Fazit und Ausblick**

Die Aktualisierung und Erweiterung der Datenbankstrukturen legt einen soliden Grundstein für die weiteren Entwicklungsarbeiten am Lernportal. Die Einführung der zusätzlichen Metadatenfelder user\_created und user\_modified ist ein klares Bekenntnis zur Transparenz und zur Verantwortlichkeit innerhalb des Systems. Mit diesen strukturellen Anpassungen ist das Projekt gut aufgestellt, um die geplanten Funktionen zur Benutzer- und Inhaltsverwaltung effektiv zu unterstützen. Die erfolgreiche Umsetzung dieser initialen Schritte bestärkt mich in meinem Vorhaben und setzt positive Impulse für die kommenden Phasen der Realisierung meines IPA-Projekts.

### Arbeitspaket 14

Beim Beginn der Realisierung des Arbeitspakets 14, stand zunächst die Herausforderung im Vordergrund, die spezifischen Felder zu identifizieren, in denen Fehler auftreten können. Diese Initialphase ist entscheidend, da sie die Basis für eine effektive Fehlerbehandlung und somit für die Sicherstellung der Datenintegrität legt.

Zur Identifizierung der Feldnamen habe ich eine Methode implementiert, die Reflection in Java nutzt, um auf die Felder des EventDTOs zuzugreifen. Reflection ermöglicht es, zur Laufzeit Informationen über Klassen und Objekte abzurufen und zu manipulieren. Im spezifischen Code-Ausschnitt:

werden alle deklarierten Felder des eventDTO-Objekts durchlaufen. Die toString()-Methode des Feldes gibt eine Zeichenkette zurück, die den vollqualifizierten Namen des Feldes enthält, was bedeutet, dass dieser String den Paketnamen gefolgt vom Feldnamen beinhaltet. Durch die Verwendung der split("\\.")-Methode wird dieser String an den Punkten aufgeteilt, wobei das Ergebnis ein Array aus String-Teilen ist. Der letzte Teil dieses Arrays (splitted[splitted.length - 1]) enthält somit den eigentlichen Namen des Feldes, frei von Paketinformationen. Diese Technik ist besonders nützlich, um dynamisch auf die Namen der Felder zuzugreifen, die für die Fehlererkennung und -behandlung relevant sind.

Die Effektivität dieses Ansatzes wurde durch Tests validiert, bei denen die Feldnamen zur Konsolen-Ausgabe von IntelliJ geleitet wurden. Dieses Vorgehen bestätigte nicht nur die Funktionalität des Codes, sondern lieferte auch die notwendige Klarheit über die Struktur der zu überwachenden Datenobjekte. Ein eingefügtes Bild der Konsolenausgabe illustriert die erfolgreiche Identifizierung der Feldnamen.

Die Verwendung dieser Technik stellt einen wichtigen Schritt im Rahmen des Arbeitspakets dar und bildet eine solide Grundlage für die weiteren Entwicklungen im Bereich der Fehlerbehandlung.

Nachdem ich erfolgreich die Feldnamen mittels Reflection ermittelt und eine initiale Struktur mittels einer switch-case-Anweisung implementiert hatte, um auf Basis dieser Feldnamen spezifische Fehlerbehandlungsmechanismen zu entwickeln, trat ich in die nächste Phase der Fehlerbehandlung ein. Diese bestand darin, die XML-Adapter so anzupassen, dass sie bei auftretenden Fehlern Exceptions werfen. Diese Exceptions sollten dann in einem übergeordneten Kontext mittels try-catch-Blöcken abgefangen werden. Bilder der Implementierung und der Versuche, die Exceptions zu fangen, dokumentieren diesen Prozess.



Unerwartet stiess ich jedoch auf ein signifikantes Problem: Die geworfenen Exceptions wurden nicht wie erwartet abgefangen. Dies führte zu einer Situation der Überforderung, da mein Plan und die vorgesehene Fehlerbehandlungsstrategie scheinbar ins Leere liefen. In solchen Momenten sind Geduld und die Bereitschaft, sich erneut in die Materie zu vertiefen und Lösungsansätze zu recherchieren, essenziell.

Mein erster Schritt bestand darin, die Ursache für das Nicht-Erfassen der Exceptions zu verstehen. Die Situation erforderte eine tiefergehende Auseinandersetzung mit der Funktionsweise von Exceptions in Java, insbesondere im Kontext von asynchronen Prozessen oder spezifischen Konfigurationen von Frameworks wie Spring Boot, welche möglicherweise Einfluss auf das Exception Handling haben könnten.

Die Recherche, gestützt durch Dokumentationen, führte mich zu einem interessanten Beitrag auf StackOverflow, <https://stackoverflow.com/questions/11114428/jaxb-xmladapter-method-does-not-throws-exception>

der die Eigenheiten von Exceptions in JAXB XMLAdaptern beleuchtet. Es wurde mir klar, dass die Art, wie diese Exceptions behandelt werden, von den normalen Java-Exceptions abweicht. Um mein Verständnis zu vertiefen und eine breitere Perspektive zu erhalten, wandte ich mich an ChatGPT mit einer spezifischen Anfrage.



**Aus dieser Interaktion ergab sich eine wichtige Erkenntnis**: Die Behandlung von Exceptions in XMLAdaptern erfordert einen spezifischen Ansatz, der sich von der herkömmlichen Java-Exception-Handhabung unterscheidet. Diese Erkenntnis war entscheidend für die Weiterentwicklung meines Projekts. Es wurde mir bewusst, dass die ursprünglich geplante Methode, benutzerdefinierte Exceptions direkt zu werfen und zu fangen, nicht die optimale Lösung für mein Problem darstellte.

Angesichts dieser Herausforderung entschied ich mich, flexibel zu bleiben und meinen Ansatz zu ändern. Ich wählte die zuvor in der Informationsphase betrachtete 2.Variante, die mit Transformations-DTOs arbeitet. Diese Entscheidung unterstreicht, wie wichtig es ist, in der Softwareentwicklung anpassungsfähig zu sein und auf unvorhergesehene Probleme mit kreativem Problemlösen zu reagieren.

Solche Herausforderungen sind Teil des Alltags in der Softwareentwicklung. Sie bieten nicht nur die Möglichkeit, unsere technischen Fähigkeiten zu schärfen, sondern erinnern uns auch daran, wie wesentlich Ausdauer, Flexibilität und die Bereitschaft, unsere Ansätze zu überdenken, für den Erfolg unserer Projekte sind.

Der erste Schritt in dem 2. Versuch der effektiven Fehlerbehandlung des Datenimports umfasste die Erstellung des EventTransformationDTO sowie der Definition verschiedener ErrorType Enums. Das EventTransformationDTO spielt eine zentrale Rolle in unserem Ansatz zur Strukturierung und Kommunikation von Fehlerinformationen. 

Innerhalb des DTOs ermöglicht die resolve Methode eine differenzierte Generierung von Ursache und Auswirkungsnachrichten, die auf dem jeweiligen ErrorType basieren. Diese sorgfältig konstruierten Nachrichten bieten Administratoren klare und präzise Fehlerbeschreibungen - von kritischen Identifikationsfehlern bis hin zu Datenkonvertierungsproblemen oder dem Fehlen essenzieller Informationen. Diese detaillierten Angaben unterstützen dabei, gezielt Korrekturen vorzunehmen und die Datenqualität zu wahren.

Darüber hinaus ermöglicht das DTO, bei Abwesenheit eines Werts und Vorhandensein eines Fallback-Wertes, eine automatische Anpassung an Standardwerte. Dieser Ansatz erhöht die Flexibilität und Stabilität unseres Systems im Umgang mit Datenimporten erheblich. Durch die Verwendung des EventTransformationDTO und der ErrorType Enums legen wir einen soliden Grundstein für eine präzise und effiziente Fehlerbehandlung, die die Integrität und Zuverlässigkeit der Datenimporte in unser System gewährleistet.

Im Zuge der Weiterentwicklung des Fehlerbehandlungsmechanismus habe ich eine signifikante Anpassung an allen XML-Adaptern vorgenommen. Die ursprüngliche Implementierung gab Optional<Integer> zurück, was zwar eine flexible Handhabung ermöglichte, jedoch nicht ideal für unsere erweiterten Anforderungen an die Fehlerbehandlung war. In der alten Version des IntegerAdapter wurde bei Fehlern lediglich ein leeres Optional zurückgegeben, ohne spezifische Informationen über den Fehler zu liefern.

Die neue Version des IntegerAdapter repräsentiert einen Paradigmenwechsel in unserer Fehlerbehandlungsstrategie.

Anstelle eines Optional<Integer> gibt der Adapter nun ein EventTransformationDTO<Integer> zurück. Dieses DTO trägt nicht nur den potenziellen Wert, sondern auch Informationen zum Fehlerzustand, falls während der Umwandlung ein Fehler auftritt. Konkret wird bei Fehlern wie fehlenden oder nicht konvertierbaren Werten der entsprechende ErrorType im DTO gesetzt. Dies ermöglicht eine deutlich präzisere Fehleranalyse und -behandlung.

Diese Anpassung gewährleistet, dass Fehler nicht einfach ignoriert werden, sondern aktiv zur Verbesserung der Datenqualität und zur effektiveren Fehlerbehandlung genutzt werden können. Die Implementierung des EventTransformationDTO und die Überarbeitung der XML-Adapter unterstreichen unser Engagement für Robustheit und Zuverlässigkeit im Umgang mit Datenimporten.

Bei der Weiterentwicklung des Projekts und insbesondere bei der Anpassung des StringAdapters und des ListAdapters stiess ich auf die Herausforderung, Konvertierungsfehler effektiv zu erkennen und zu handhaben. Um eine präzise Fehlererkennung zu ermöglichen, wurde mir klar, dass der bestehende StringCleaner angepasst werden muss, um nicht nur die Daten zu bereinigen, sondern auch potenzielle Fehler während der Konvertierung zu identifizieren und entsprechend zu reagieren.

Um dieses Ziel zu erreichen, entschied ich mich für die Implementierung einer spezifischen ConversionException.   
  
Diese massgeschneiderte Exception ermöglicht eine gezielte Behandlung von Konvertierungsfehlern, die während der Datenbereinigung durch den StringCleaner auftreten können. Die Anpassung des StringCleaners beinhaltete die Einführung von Try-Catch-Blöcken rund um die Datenverarbeitungslogik, wodurch bei Auftreten eines Fehlers eine ConversionException ausgelöst wird. Diese methodische Anpassung erlaubt es, Fehler sofort zu erkennen und entsprechende Massnahmen zu ergreifen, um die Integrität und Zuverlässigkeit der Datenkonvertierung zu gewährleisten.

Bei diesem Vorgehen kommt eine try-catch-Struktur zum Einsatz. Während der try-Block die Reinigungs- und Konvertierungslogik des StringCleaners ausführt, ist der catch-Block darauf ausgerichtet, jegliche ConversionExceptions, die im Prozess entstehen, aufzufangen. Sobald eine solche Exception gefangen wird, setzt der Adapter im EventTransformationDTO den Fehlerstatus auf ErrorType.CONVERSION. Dieser Mechanismus gewährleistet, dass Konvertierungsfehler effizient identifiziert und entsprechend markiert werden, was eine klare Fehlerkommunikation und -behandlung innerhalb des Datenverarbeitungsflusses ermöglicht.

Diese Methode ist der Kern der Anpassungen in der MyTalentTransformer-Klasse, die es ermöglicht, Fehler effizient zu identifizieren und zu behandeln, sobald Daten aus den angepassten XML-Adaptern kommen.

Diese Methode nimmt ein EventTransformationDTO, die ID des Events, den Titel des Events und den Namen des Feldes, bei dem der Fehler aufgetreten ist, entgegen. Die Logik innerhalb von setValueCheckErrors prüft zunächst, ob ein Fehler vorliegt (errorType ist nicht null). Abhängig vom Feldnamen werden dann spezifische Aktionen durchgeführt, um sicherzustellen, dass keine kritischen Informationen fehlen. Zum Beispiel wird, falls der Titel des Events fehlt, ein Standardtitel "Veranstaltung ohne Titel" zugewiesen. Dieser Ansatz ermöglicht es, die Datenqualität zu wahren und bietet gleichzeitig eine klare Dokumentation der Fehlerursachen und ihrer Auswirkungen.

Nachdem die erforderlichen Korrekturen und Anpassungen vorgenommen wurden, werden die ID und der Titel des Events im DTO gesetzt, und die resolve-Methode generiert präzise Fehlermeldungen basierend auf dem Fehlertyp. Anschliessend werden alle relevanten Fehlerdaten mittels eventService.createTransformations in die Datenbank eingetragen. Dies geschieht gemäss dem etablierten Schema des Projekts und gewährleistet eine konsistente und nachvollziehbare Speicherung von Fehlerinformationen.

Die Herausforderungen, insbesondere das Problem mit der Handhabung von Exceptions in den XML-Adaptern, machten diesen Prozess nicht leicht. Jedoch war es entscheidend, ruhig zu bleiben und nach alternativen Lösungen zu suchen. Die Entscheidung, auf das EventTransformationDTO und die ErrorType-Enums zu setzen, ermöglichte eine elegante und übersichtliche Fehlerbehandlung. Es zeigte sich wieder, wie wichtig es ist, flexibel auf unvorhergesehene Probleme zu reagieren und den Entwicklungsprozess entsprechend anzupassen.

Um den Abschluss meines Arbeitspaketes 14 zu dokumentieren, habe ich eine wesentliche Anpassung im Importprozess vorgenommen. Statt direkt EventsXML als Antwort von meinem Import zu erhalten, habe ich eine neue Struktur eingeführt, die als EventImportDTO bezeichnet wird. Dieses Data Transfer Object umfasst nicht nur das EventsXML-Objekt selbst, sondern hält auch den Importstatus fest, der aus dem Ergebnis des Anfrageprozesses resultiert.

Im Kern der Implementierung steht die MyTalentClient-Klasse, die für die Anfrage an die externe API verantwortlich ist. Mittels eines RestTemplate wird die Anfrage gestellt, und je nach Antwortstatus des Servers wird der Importstatus im EventImportDTO gesetzt. Dies erlaubt es, differenzierte Rückmeldungen über den Erfolg des Imports zu geben, sei es ein Netzwerkproblem, ein allgemeiner Fehler oder ein erfolgreicher Import.

Nach Erhalt der Antwort wird das EventImportDTO in die Datenbank eingespeichert, wobei ich sorgfältig darauf geachtet habe, das etablierte Schema des Projekts zu befolgen. Diese Praxis gewährleistet eine konsistente Datenstruktur und erleichtert die nachfolgende Datenverarbeitung und -analyse.

Durch die Implementierung des EventImportDTO und die strukturierte Anpassung der Fehlerbehandlung konnte ich nicht nur die Datenintegrität und -verarbeitung signifikant ver-bessern, sondern auch eine robuste Grundlage für die zukünftige Erweiterbarkeit des Sys-tems schaffen. Diese Entwicklungsphase hat gezeigt, dass es entscheidend ist, flexibel zu reagieren und konstruktive Lösungsansätze für unvorhergesehene Probleme zu entwickeln.

**Fazit Arbeitspaket Errorhandling**

Das Fazit zu Arbeitspaket 14 fasst einen wesentlichen Entwicklungsschritt in der Fehlerbehandlung und Datenverarbeitung meines Projekts zusammen. Herzstück der Anpassung ist die Überarbeitung in der MyTalentTransformer-Klasse, welche zuvor durch eine unelegante und schwer zu wartende Implementierung gekennzeichnet war. Die Einführung einer umfangreichen switch-Case-Anweisung ermöglicht nun eine klare Identifizierung von Feldnamen und eine strukturierte Fehlerbehandlung.

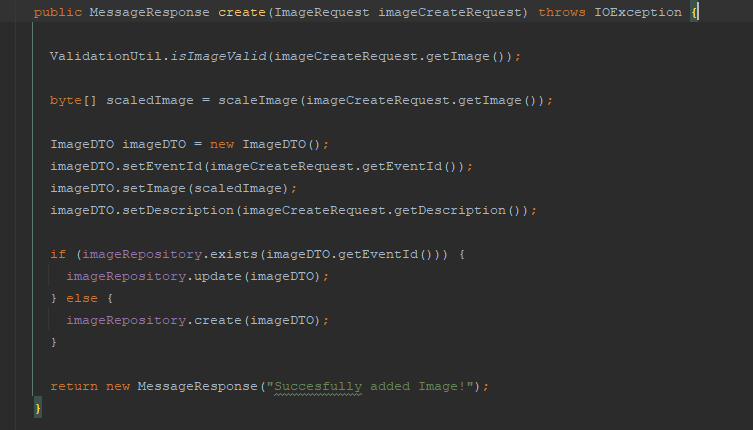
Das grösste Problem, dem ich mich gegenübersah, betraf die Herausforderungen mit den XML-Adaptern. Diese Schwierigkeit hat deutlich gemacht, wie entscheidend es ist, auch unter Druck einen kühlen Kopf zu bewahren. Statt mich von der ersten Hürde entmutigen zu lassen, habe ich mich intensiv mit dem Problem auseinandergesetzt und durch gründliche Recherche und Anpassungsfähigkeit eine praktikable Lösung gefunden. Diese Erfahrung unterstreicht die Bedeutung von Ausdauer und einem methodischen Ansatz in der Softwareentwicklung.

## Arbeitspaket 15

Für das Arbeitspaket 15, die Entwicklung der Bilderverwaltungsfunktionen, begann ich mit der Aktualisierung der pom.xml und application.properties, um die notwendigen Abhängigkeiten und Konfigurationen für die Verwendung von Swagger für das Testen der Endpunkte hinzuzufügen. Nachdem die Grundlagen geschaffen waren, richtete ich meine Aufmerksamkeit auf den Aufbau der grundlegenden Struktur, die aus einem ImageController, einem ImageService und einem ImageRepository besteht. Diese Struktur ermöglichte es mir, mit der Implementierung der geforderten Funktionen zu beginnen.

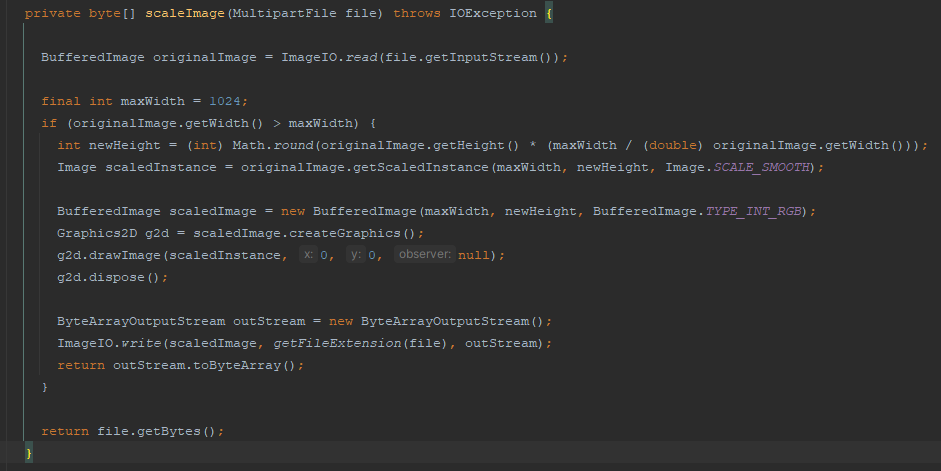
Nachdem die Struktur etabliert war, habe ich mir die Aufgabenstellung noch einmal genau durchgelesen, um sicherzustellen, dass ich alle Anforderungen verstehe. Die Aufgabenstellung umfasste die Erstellung von Endpunkten für das Erstellen, Aktualisieren, Löschen und Laden von Bildern. Daraufhin habe ich diese Endpunkte entsprechend den Vorgaben und Standards meiner Firma implementiert. Besondere Aufmerksamkeit widmete ich dabei der Entscheidung, die eventId als Primärschlüssel in der Bildtabelle zu verwenden, anstatt einer UUID. Dies war eine bewusste Entscheidung, um die Konsistenz zu gewährleisten das ein Event nur ein Bild haben kann.

Bei der Implementierung des Create-Endpoints fügte ich eine Logik hinzu, die zunächst prüft, ob bereits ein Eintrag mit der gegebenen eventId existiert.

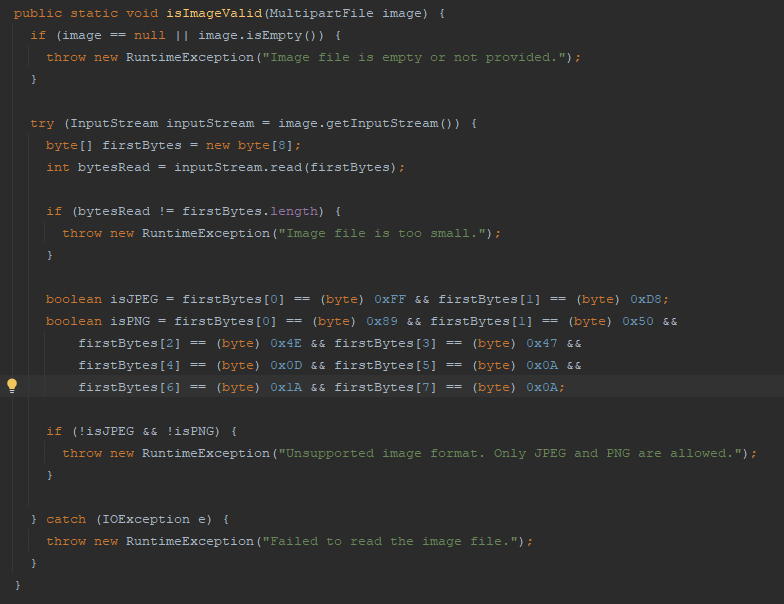


Falls ein solcher Eintrag gefunden wird, dessen is\_deleted-Flag auf true gesetzt ist, wird dieser Eintrag durch Setzen des Flags auf false wieder aktiviert. Diese Herangehensweise ermöglicht es, Daten effizient wiederzuverwenden und unnötiges Anlegen von Duplikaten zu vermeiden. Die Implementierung der anderen Endpunkte folgte ähnlichen Überlegungen, wobei ich sicherstellte, dass alle Operationen den Anforderungen entsprechend funktionieren und in das Gesamtschema der Bilderverwaltung passen.

Für die Skalierung der Bilder orientierte ich mich an den Recherchen und Empfehlungen von Baeldung Java Bildskalierung Dokumentation und entschied mich für den Einsatz von Graphics2D, einer Klasse aus der Java-Standardbibliothek.

Die Methode scaleImage im ImageService nimmt eine Bilddatei als Eingabe und prüft, ob die Breite des Bildes die maximale Breite von 1024 Pixeln überschreitet.   
  
Ist dies der Fall, wird das Bild proportional skaliert, sodass die maximale Breite eingehalten wird, während die Bildqualität durch die Verwendung von Image.SCALE\_SMOOTH erhalten bleibt. Anschliessend wird das skalierte Bild in einem BufferedImage gespeichert und schliesslich in ein Byte-Array umgewandelt, das zurückgegeben wird. Diese Herangehensweise stellt sicher, dass die Bilder in einer einheitlichen Grösse gespeichert werden, was für eine konsistente Darstellung und effiziente Speicherung sorgt.

Beim Umgang mit Bilduploads in meinem Projekt stellte die Validierung der Bilddateien eine besondere Herausforderung dar. Ziel war es, sicherzustellen, dass hochgeladene Dateien tatsächlich den unterstützten Bildformaten JPEG oder PNG entsprechen. Um eine möglichst hohe Sicherheit zu gewährleisten und gleichzeitig die Abhängigkeit von externen Bibliotheken zu minimieren, entschied ich mich gegen die Nutzung des MIME-Typs als alleiniges Kriterium für die Validierung. Stattdessen entwickelte ich eine Methode, die direkt die ersten Bytes einer Datei analysiert.



Die Methode isImageValid prüft die ersten 8 Bytes der Bilddatei gegen die spezifischen Signaturen für JPEG- und PNG-Dateiformate. Diese Herangehensweise bietet den Vorteil einer zuverlässigen und sicheren Erkennung des Bildformats, da diese Signaturen eindeutig sind und nicht manipuliert werden können, ohne die Datei unbrauchbar zu machen. Ein weiterer entscheidender Vorteil dieser Lösung ist, dass sie eine zusätzliche Abhängigkeit von umfangreichen Bibliotheken wie Apache Tika vermeidet. Obwohl Tika eine umfassende Lösung für die Dateityp-Erkennung bietet, war es mein Ziel, die Projektstruktur so schlank und unabhängig wie möglich zu halten. Durch die Implementierung einer eigenen Validierungsmethode, die sich auf grundlegende, aber effektive Prüfungen stützt, konnte ich dieses Ziel erreichen und gleichzeitig die Sicherheit und Integrität des Bildupload-Prozesses gewährleisten.

Diese Arbeit an der Bilderverwaltung war nicht nur technisch anspruchsvoll, sondern gab mir auch die Möglichkeit, tief in die Anforderungen und die beste Herangehensweise für deren Umsetzung einzutauchen. Durch die detaillierte Planung und das methodische Vorgehen konnte ich eine solide Lösung schaffen, die den Bedürfnissen des Projekts gerecht wird.

### Arbeitspaket 16 & 17

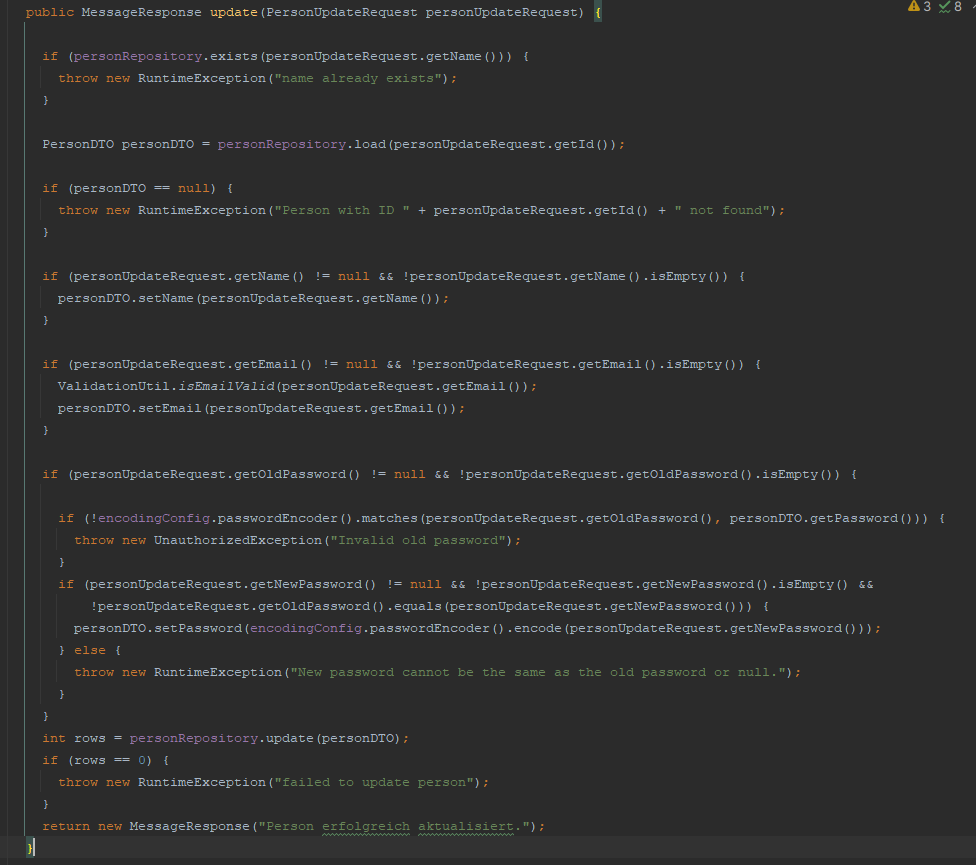
## Beim Implementieren der Arbeitspakete 16 und 17 für die Sicherheitsmechanismen und die Benutzerverwaltung im Backend, habe ich mich an ein bestehendes Spring Security Schema eines anderen Projekts meiner Abteilung orientiert. Diese Referenz diente als wertvolle Grundlage, jedoch musste ich feststellen, dass eine direkte Übernahme nicht möglich war. Der Grund dafür lag in den unterschiedlichen Anforderungen: Während das Referenzprojekt auf einer rollenbasierten Zugriffskontrolle aufbaute, sind in meinem Projekt keine spezifischen Benutzerrollen definiert. Diese Erkenntnis führte dazu, dass ich die Spring Security Konfiguration grundlegend anpassen und für die spezifischen Bedürfnisse meines Projekts optimieren musste.

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

Automatisch generierte BeschreibungIm Rahmen der Benutzerverwaltung war es mir ein Anliegen, nicht nur eine effiziente, sondern auch eine intuitive Lösung zu schaffen, die es den Benutzern ermöglicht, Benutzerkonten einfach zu verwalten. Neben den grundlegenden Funktionen zur Erstellung und Bearbeitung von Benutzerkonten, habe ich einen flexiblen Ansatz für die Aktivierung und Deaktivierung von Benutzerkonten implementiert.

Hierfür entwickelte ich einen spezifischen Endpunkt, der sowohl für die Aktivierung als auch für die Deaktivierung von Benutzerkonten verwendet werden kann. Diese flexible Lösung ermöglichte es mir, die Entwicklung eines separaten Endpunkts für diese Funktionen zu umgehen. Stattdessen kann im Frontend einfach die entsprechende Eingabe gemacht werden, je nachdem, ob ein Benutzer aktiviert oder deaktiviert werden soll. Diese Herangehensweise spart nicht nur wertvolle Entwicklungszeit, sondern trägt auch zur Vereinfachung der Verwaltung und Wartung der Anwendung bei.

Darüber hinaus nahm ich eine Verfeinerung der Aufgabenstellung vor, die ursprünglich nicht die Möglichkeit vorsah, deaktivierte Benutzer wieder zu aktivieren. Nach meiner Einschätzung ist diese Funktion jedoch essenziell, insbesondere im Hinblick auf die zukünftige Nutzung des Lernportals durch HR-Mitglieder, die entweder Kurse mit Bildmaterial bestücken oder Fehler des Importvorgangs einsehen möchten. Diese Anpassung erweitert die Flexibilität und Funktionalität des Systems erheblich und stellt sicher, dass es den sich entwickelnden Anforderungen der Benutzer gerecht wird.

Die update-Methode für Personendaten repräsentiert einen sorgfältig durchdachten Prozess, der sowohl Funktionalität als auch Benutzerfreundlichkeit im Blick hat. Im Folgenden dokumentiere ich die Schlüsselkomponenten und Überlegungen, die in die Gestaltung dieser Methode eingeflossen sind, um eine effiziente und sichere Aktualisierung von Personendaten zu gewährleisten.

**Überprüfung der Einzigartigkeit des Namens**

Der erste Schritt im Update-Prozess ist die Überprüfung, ob der neue Name bereits im System vorhanden ist. Dies verhindert Duplikate und gewährleistet, dass jeder Benutzer im System einen einzigartigen Namen hat. Diese Prüfung ist entscheidend für die Integrität der Daten und vermeidet Verwirrung oder Konflikte bei der Identifikation von Personen.

**Überprüfung der Existenz der Person**

Bevor Änderungen vorgenommen werden, stellt die Methode sicher, dass die zu aktualisierende Person anhand der übergebenen ID tatsächlich existiert. Dies schützt vor fehlerhaften Datenoperationen und stellt sicher, dass nur vorhandene Datensätze bearbeitet werden. Die klare Fehlermeldung, die ausgegeben wird, wenn keine Übereinstimmung gefunden wird, dient der besseren Nachvollziehbarkeit und Benutzerführung.

**Flexible Aktualisierung von Datenfeldern**

Die Methode erlaubt die selektive Aktualisierung von Namen und E-Mail-Adressen. Indem nur Felder aktualisiert werden, die nicht null und nicht leer sind, bietet die Methode Flexibilität und Benutzerfreundlichkeit. Benutzer können einzelne Informationen ändern, ohne alle Daten neu eingeben zu müssen. Die Überprüfung der E-Mail-Adresse auf Gültigkeit, bevor sie aktualisiert wird, erhöht zusätzlich die Datenqualität.

**Sichere Handhabung von Passwörtern**

Die Methode behandelt die Aktualisierung von Passwörtern mit besonderer Sorgfalt. Zunächst wird überprüft, ob das alte Passwort korrekt ist, was ein wichtiges Sicherheitsmerkmal darstellt, um unberechtigte Änderungen zu verhindern. Anschliessend wird sichergestellt, dass das neue Passwort nicht gleich dem alten ist und nicht leer ist. Die Verwendung eines Passwort-Encoders zur Speicherung des neuen Passworts in verschlüsselter Form ist ein weiterer entscheidender Schritt zum Schutz der Benutzerdaten.

**Fehlerbehandlung und Rückmeldung**

Die Methode ist so konzipiert, dass sie im Fehlerfall aussagekräftige Nachrichten zurückgibt. Dies umfasst sowohl die Handhabung von Situationen, in denen die Aktualisierung aufgrund von Datenvalidierungsfehlern nicht durchgeführt werden kann, als auch die Bestätigung des Erfolgs, wenn die Aktualisierung erfolgreich war. Diese klaren Rückmeldungen verbessern das Benutzererlebnis, indem sie dem Benutzer eindeutige Hinweise zum Status der Operation geben.

**Fazit**

Die update-Methode demonstriert einen durchdachten Ansatz zur Handhabung von Personendaten-Aktualisierungen, der Sicherheit, Benutzerfreundlichkeit und Datenintegrität in den Vordergrund stellt. Durch die Berücksichtigung von Benutzererfahrungen bei jeder Entscheidung, von der Validierung der Eingaben bis zur detaillierten Fehlerbehandlung, wird eine solide Grundlage für zuverlässige und benutzerzentrierte Datenoperationen geschaffen.

**Die** **Implementierung der Sessionverwaltung** war ein zentraler Bestandteil meines Projekts, um eine sichere Benutzeranmeldung und -abmeldung zu gewährleisten. Ich entwickelte das AuthenticationRepository und den AuthenticationService, die sich nahtlos in das bestehende Schema meiner Services, wie dem PersonService oder dem ImageService, integrieren. Im AuthenticationService wurden die Prozesse für das Einloggen und Ausloggen von Benutzern sorgfältig abgebildet und umgesetzt.

Ein Schlüsselaspekt dieser Implementierung ist der sorgfältige Umgang mit Benutzersessions.

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Beim Login eines Benutzers prüft der AuthenticationService zunächst, ob bereits eine gültige Session für die betreffende Person existiert. Sollte dies der Fall sein, wird die existierende Session aus Sicherheitsgründen gelöscht und eine neue Session erstellt. Dieser Ansatz verhindert effektiv, dass Benutzer mehrere aktive Sessions gleichzeitig unterhalten können, wodurch das Risiko unbefugter Zugriffe minimiert wird.

Zur Verbesserung der Sicherheit habe ich zudem eine UnauthorizedException eingeführt. Diese Ausnahme wird geworfen, wenn ein Benutzer mit ungültigen Anmeldeinformationen versucht, sich einzuloggen, und dient als Schutzmechanismus gegen unautorisierte Zugriffsversuche. Durch diese strategische Implementierung und das Hinzufügen spezifischer Sicherheitsausnahmen konnte ich die Authentizität und Sicherheit der Benutzersessions signifikant erhöhen und somit die Grundlage für ein vertrauenswürdiges und sicheres Backend-System schaffen.

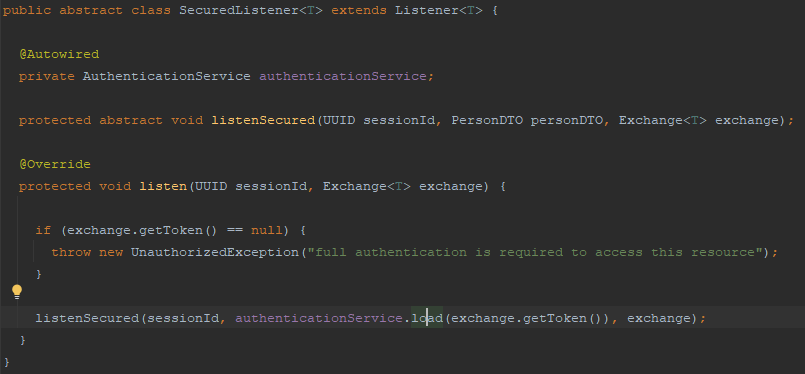
**Herausforderung: Sicherheit im WebSocket-Bereich**

Die erste grosse Herausforderung des Tages war die Implementierung eines Sicherheitsmechanismus für den WebSocket-Bereich. Die Kernfrage dabei war: Wie gestalte ich den Sicherheitsbereich effektiv? Speziell ging es darum, an welcher Stelle der Authentifizierungscheck stattfinden sollte – beim Listener oder beim Responder. Die Überlegung war, den Prozess so elegant wie möglich zu gestalten, um Redundanzen im Code zu vermeiden.

**Lösungsansatz: Secured WebSocket Listener**

Nach eingehenden Überlegungen entschied ich mich, den Token-Check direkt beim WebSocket-Listener zu integrieren. Der Grund dafür ist, dass bevor eine Interaktion mit dem Endpunkt stattfinden darf, gewährleistet sein muss, dass der Benutzer über einen gültigen Token verfügt. Die Frage war nun, wie der Authentifizierungscheck auf elegante Weise implementiert werden kann, ohne redundante Codeabschnitte zu erzeugen.

Die Lösung fand ich in der Erstellung eines neuen Typs von Listener – dem SecuredListener. Dieser führt den Token-Check durch, den ich zuvor im Rahmen der Implementierung der Sessionverwaltung in das AuthenticationService integriert hatte.

Implementierung des SecuredListeners:

Der SecuredListener erweitert einen allgemeinen Listener und fügt eine Sicherheitsebene hinzu, indem er vor der eigentlichen Verarbeitung der Nachricht überprüft, ob ein gültiges Token vorliegt. Dies ermöglicht eine effiziente und sichere Handhabung der WebSocket-Kommunikation.

Die Entscheidung, den Sicherheitscheck in den SecuredListener zu integrieren, erwies sich als effektiv.

### Arbeitspaket 18 und 2. Meilenstein

Im Rahmen des Feinschliffs des Backends für meine IPA habe ich einen klaren Plan verfolgt, um das Backend meiner Anwendung zu optimieren und zu verfeinern. Mein Ansatz umfasste mehrere zentrale Schritte, die darauf abzielten, die Funktionalität zu erweitern, den Code zu refaktorisieren und letztlich die Anwendung zu verbessern, ohne dabei die Qualität oder die Sicherheit zu kompromittieren.

**Implementierung fehlender Funktionen**

Zunächst konzentrierte ich mich darauf, alle notwendigen Funktionen zu implementieren, die für eine vollständige Backend-Lösung erforderlich waren. Dazu gehörten insbesondere neue Endpunkte zum Laden der Transformationen sowie Details zu den jeweiligen Bildern der Events. Diese Erweiterungen waren entscheidend, um die Benutzererfahrung zu bereichern und die Anwendung umfassender und interaktiver zu gestalten.

**Code-Refactoring**

Nachdem die wesentlichen Funktionen implementiert waren, widmete ich mich dem Refactoring des Codes. Mein Ziel war es, den Code zu straffen, Redundanzen zu eliminieren und die Lesbarkeit sowie die Wartbarkeit zu verbessern. Ein besonderes Augenmerk legte ich dabei auf das ImageRepository, wo ich die Nutzung von User-Created- und User-Modified-Metadaten mithilfe der SessionUtil-Klasse optimierte. Durch diese Anpassungen konnte ich die Codequalität signifikant steigern und gleichzeitig die Funktionalität der Anwendung erweitern.

**Optimierung des Transformationsprozesses**

Ein weiterer wichtiger Punkt meines Feinschliffs war die Optimierung des Transformationsprozesses. Nach einem produktiven Austausch mit Flo im Daily habe ich verschiedene Möglichkeiten zur Verbesserung dieses Prozesses erörtert. Trotz meines starken Wunsches, den MyTalentTransformer für eine bessere Übersichtlichkeit und Effizienz zu überarbeiten, musste ich erkennen, dass unter dem gegebenen Zeitdruck Prioritäten gesetzt werden mussten.

**Entscheidungen unter Zeitdruck**

Obwohl der MyTalentTransformer auf den ersten Blick als komplex und unübersichtlich erscheinen mag, war es mir wichtig zu betonen, dass bereits erhebliche Verbesserungen erzielt wurden. Die Herausforderung, eine grosse Menge an Daten und Feldern zu verarbeiten, war enorm, und ich bin stolz auf das, was ich in dieser kurzen Zeit erreichen konnte. Dennoch bin ich mir bewusst, dass mit mehr Zeit und Ressourcen eine weitere Optimierung des Transformationsflusses möglich wäre, etwa durch die Einführung spezifischer Adapter für Speaker und Kategorien, die jeweils den StringCleaner integrieren würden. Diese Änderungen könnten die Notwendigkeit der Methoden addSpeaker, findCategory, isCategory undconvertSubcategory eliminieren und den Code im Transformer weiter vereinfachen.

**Erkenntnisse und strategische Entscheidungen bezüglich des StringCleaner**

Durch den wertvollen Input meiner Verantwortlichen Fachkraft Flo wurde mir bewusst, dass die ursprünglich in den StringAdapter und ListAdapter implementierten try-catch-Blöcke für die ConversionException im Umgang mit dem StringCleaner unter den aktuellen Umständen nicht notwendig waren. Dies lag daran, dass im StringCleaner keine Exceptions geworfen wurden, ausser der übergebene Wert war null. In jedem Adapter, in dem der StringCleaner verwendet wurde, erfolgte bereits eine Überprüfung auf null, bevor der StringCleaner zum Einsatz kam. Diese Erkenntnis könnte als "unnötig" angesehen werden. Ursprünglich dachte ich, dass diese Blöcke erforderlich seien, um potenzielle Konvertierungsfehler abzufangen. Jedoch stellte sich heraus, dass sowohl im StringAdapter als auch im ListAdapter tatsächlich keine Konvertierungsfehler auftreten können, da alle Eingaben als Strings empfangen und verarbeitet werden. Dieser Logikfehler wurde im Zuge des Feinschliffs korrigiert und die unnötigen try-catch-Blöcke wurden entfernt. Diese Entscheidung vereinfachte den Code erheblich und verbesserte die Klarheit und Effizienz der betroffenen Komponenten.

**Fazit**

Diese Phase des Feinschliffs war geprägt von kontinuierlichem Lernen und Anpassung. Durch den proaktiven Dialog mit Flo und die Bereitschaft, meine Annahmen zu hinterfragen, konnte ich nicht nur die Implementierung verbessern, sondern auch mein Verständnis für effektive Fehlerbehandlung und Code-Optimierung vertiefen. Mit diesen Anpassungen habe ich einen wichtigen Schritt in Richtung einer optimierten, benutzerfreundlichen Backend-Lösung gemacht, die einen soliden Grundstein für die zukünftige Entwicklung legt.

### Arbeitspaket 19

t

## Phase "Kontrollieren"

## Phase "Auswerten"

## Persönliches Fazit

## Quellenverzeichnis

## Glossar

|  |  |
| --- | --- |
| Begriff | Definition |
|  |  |
|  |  |