IPA Marc Egli - Puzzle ITC

IPA-Daten und beteiligte Personen	
Firma, Abteilung	Puzzle ITC, /dev/ruby
Berufsschule	GIBB
Valid Experte	Lawson Mike
Hauptexpertin	Müller Lorenz
Nebenexperte	Moser Michael
Verantwortliche Fachkraft	Illi Daniel
Zusätzliche Verantwortliche Fachkraft	Steiner Robin
Berufsbildner	Steiner Robin
Fachrichtung	Applikationsentwicklung
Projektvorgehensmodell	SCRUM
Jahrgang der IPA-Durchführung und Kanton	IPA 2025, Kanton Bern
Abgabedatum	22.01.2025

Tabelle 1: IPA Daten

Teil I

Ablauf, Organisation und Umfeld

Inhaltsverzeichnis

1 Aufgabenstellung

1.1 Titel der Arbeit

Hitobito: Neue Generation von Personen-Filtern

1.2 Thematik

Eines der Kernfunktionalitäten von Hitobito ist das Filtern via vom Benutzer definierten Kriterien von Personen auf Personenlisten und Abos. Diese Funktionalität ist in den über 10 Jahren seit es Hitobito gibt oft erweitert worden. Durch die vielen neuen Filtermöglichkeiten wurde speziell das UI immer komplexer und unübersichtlicher. Die Personen-Filteroptionen für Personenlisten und die der Abos sehen ähnlich aus, weisen aber diverse nicht offensichtliche Unterschiede auf. Mit dieser Probe-IPA soll für den Backendteil der Abos (MailingLists) eine neue Generation von Personen-Filtern für Hitobito entwickelt werden.

1.3 Ausgangslage

Hitobito ist eine Open Source Webapplikation zum Verwalten von Mitgliedern, Events und vielem mehr. Die Ruby on Rails Applikation wurde 2012 von Puzzle ITC initiiert und wird stets weiterentwickelt.

Die Basis für die Software bildet das Webframework Ruby on Rails. Für das User Interface wird neben statischer Technologie wie HTML und CSS auch JavaScript oder Hotwire verwendet. Der komplette Source-Code steht auf Github zur Verfügung: https://github.com/hitobito

1.4 Detaillierte Aufgabenstellung

Mit dieser Probe-IPA soll ein neues Konzept und Datenmodell für die Persistierung von Filter-Parametern erstellt werden (rein Backend). Anschliessend soll dieses Konzept in einem Proof of Concept (PoC) bei einem Teil der Mailinglisten (Abos) umgesetzt werden.

- Die Klassen Subscription, RelatedRoleType, PeopleFilter, usw. werden im neuen Konzept komplett ersetzt oder ggf. ergänzt
- Eine Möglichkeit ist das PeopleFilter die Basis für das neue Konzept bilden
- Es sollen 2-3 Grobkonzepte gegenüber gestellt werden und das ausgewählte Konzept detaillierter ausgearbeitet werden

PoC

- Folgende Komponenten der MailingLists Filter sollen mit dem neuen Konzept im PoC umgesetzt werden:
  - Globale Bedingungen ¿ Sprache
  - Personen
  - Ausgeschlossene Personen
  - Optional: Gruppen / Rollen
- Persistierte Subscriptions/Filter müssen für den PoC vorerst nicht migriert werden
- Die nicht erwähnten Komponenten müssen nicht mehr funktionieren
- Die erwähnten Komponenten (ohne Optionale) funktionieren im UI und haben eine minimale, funktionierende Testabdeckung (happy path)

Out of Scope - wird nicht oder erst nach der Probe IPA umgesetzt

- Konzept und Anpassungen Frontend/UI
- PoC Umbau/Migration People Filter Personenlisten
- JSON API Filter (Grafiti)

1.4.1 Mittel und Methoden

Technologie und Plattform:

- Ruby, Ruby on Rails, Active Record

Entwicklungsumgebung:

- Intellij
- Git, Github
- Rake
- Rubocop

Textverarbeitung und Diagramme:

- Latex
- draw.io
- Google Sheets

Projektmethode:

- Scrum IPA

Konventionen:

- Es gilt der Ruby Style Guide und der Rails Style Guide gemäss Rubocop Konfiguration des Projekts

1.4.2 Vorkenntnisse

Marc arbeitet bereits seit einigen Monaten an Features von Hitobito. Ausserdem hat er bereits seit dem 2. Lehrjahr Erfahrung auch in anderen Ruby on Rails Projekten gesammelt.

1.4.3 Vorarbeiten

- Vorbereitung Dokumentvorlage
- Ist-Analyse Personen-Filter Personen-Listen/Abos
- Dokumentation in der Developer-Dokumentation der bestehenden Implementation von MailingLists, FilteredList, Personen-Filter

1.4.4 Neue Lerninhalte

- Eigenständiges Entwerfen der Datenstruktur/Klassen

1.4.5 Arbeiten in den letzten 6 Monaten

- Umsetzung diverser Features für Hitobito (Ruby on Rails)
- Postgresql Migration Hitobito

2 Firmenstandards

2.1 Code conventions

Als Code convention werden die Ruby Style Guides verwendet. Die Überprüfung dieser Style Guidelines wird mit Rubocop (Formatter) sichergestellt. Die Konfiguration dieses Formatters ist unter rubocop.yml ersichtlich.

2.1.1 Lizenz

In jedem File in Hitobito wird das Copyright für den jeweiligen Kunden und die Lizenz dazu in Kommentarform beschrieben. Diese Lizenz- sowie Kundeninformationen können über folgenden Befehl eingefügt werden.

```
rake license:insert
```

Alternativ dazu können diese Informationen mit

```
rake license:remove
```

entfernt oder mit

```
rake license:update
```

aktualisiert werden.

2.2 Git conventions

Für das cloudbasierte Hosting unseres Git-Repositories wird Github verwendet. Die Git Commitnachrichten werden nach den Regeln von Puzzle ITC formuliert. Im Anhang unter Git Conventions finden sie eine Kopie unserer Firmenkonventionen

- Sprache: Englisch
- Kurze und prägnante Message, idealerweise unter 50 Zeichen Details
- Mit Grossbuchstaben beginnen Details
- Kein Punkt am Schluss Details
- Den *imperative mood* (Befehlsform) verwenden, also «Fix bug with X» statt «Fixed bug with X» oder «More fixes for broken stuff» Details
- Wenn vorhanden Ticket referenzieren:
  - Bei Open Project Work Packages: «Add X, refs #12345»
  - Bei Gitlab/Github Issues: «Add X #12345»

2.3 Documentation Conventions

Als Documentation covention wird arc42 verwendet (Siehe arc 42 documentation).

2.4 Security Conventions

- Injection / Cross Size Scripting
  - Input Validierung von allen Inputs serverseitig durchführen
  - Output Encoding auf allen Outputs anwenden
  - Kein inline oder dynamisches SQL, sondern parametrisierte Queries verwenden
  - Datei Uploads überprüfen
- Verbindungs- / Browsersicherheit
  - Nur HTTPS verwenden und korrekt konfigurieren
  - Security Headers setzen
  - Cookie Flags secure, httpOnly und SameSite setzen
  - Kein Caching von sensiblen Informationen
- Authentication / Sessions
  - IAM des Frameworks oder besser Keycloak verwenden
  - Keine sensitiven Infos in URL Parameter
  - Brute Force Schutz
  - Sessions schützen
- Tools und Betriebsumgebung
  - Errorhandling und Logging
  - Libraries und deren Dependencies auf bekannte Schwachstellen prüfen
  - OS, Webserver, Container aktuell halten und Hardening
  - Keine Produktionsdaten auf Integrationsumgebungen
- Security Testing
  - Es dürfen keine Secrets im Repository abgelegt werden
  - Eingebundene Dependencies dürfen keine MEDIUM und HIGH Schwachstellen aufweisen
  - Eine statische Codeanalyse sollte durchgeführt werden
  - Eine dynamische Codeanalyse sollte durchgeführt werden
  - Alle verwendeten Images sollten auf Schwachstellen gescannt werden

3 IPA-Schutzbedarfanalyse

3.1 Datensicherheit

Die notwendigen Daten welche im Rahmen der IPA zu Test- und Vorführungszwecken verwendet werden, sind werden durch das Faker-Gem generiert und sind somit NICHT besonders schützenswert. Dazu gehören unter anderem Adressen, Familiendaten, Finanzdaten.

3.2 Applikationssicherheit

Obwohl im Rahmen der IPA nicht mit besonders schützenswerten Daten gearbeitet muss bei der Programmierung beachtet werden, dass bei den Filterungen stets auf ein Datenset zugegriffen wird welches durch das can-can-can-Gem validiert wurde um zu Verhindern dass Personen auf die Daten anderer Zugriff haben. Dies ist wichtig, da sich bei späterer Implementierung der IPA in Hitobito besonders Schützenswerte Daten in der Datenbank befinden. Das Datenschutzkonzept dafür finden sie

4 Organisation der IPA-Ergebnisse

4.1 Datensicherung

In dieser IPA unterteilen wir die Datensicherung in:

- Dokumentation
- Code

4.1.1 Dokumentation

Dokumentation	
Tools	Git und USB
Versioniert	Ja
Interval	Mind. 2x täglich
Beschreibung	Die Dokumentation ist im ipa-puzzle-template Repository unter dem Branch probe-ipa angelegt. Sobald ein Dokumentationsticket abgeschlossen wurde, werden die Änderungen auf den Github Server in das private Repository gepushed. Dies geschieht mind. 2x täglich. Zusätzlich, wird pro Tag ein Ordner auf einem USB-Stick erstellt. Am Ende des Tages wird eine Kopie der Dokumentation in diesen Ordner geladen.

Tabelle 4.1: Sicherung Dokumentation

4.1.2 Code

Code	
Tools	Git und USB
Versioniert	Ja
Interval	Mind. 2x täglich
Beschreibung	Für die Entwicklung habe wurden die Repositories hitobito und hitobitogeneric geforked. Auf diesen Repositories wird an Tagen an welchen Entwickelt wird, mind. 2x täglich committed. An diesen Tagen wird zur doppelten Sicherung zusätzlich eine Kopie des Projektes auf den USB Stick gespeichert, unter dem Ordner des jeweiligen Tages.

Tabelle 4.2: Sicherung Code

4.1.3 Wiederherstellung des Codes

Gehen die Daten lokal verloren, können diese entweder über das Github Repository oder den USB-Stich wiederhergestellt werden. Bei der Wiederherstellung mit Git, wird der SSH-Key des Repositories benötigt, damit dieses von Github geklont werden kann. Ist dieser SSH-Key nicht verfügbar, wird die Wiederherstellung über den USB-Stick vorgenommen und das Projekte des letzten Speicherstandes kopiert. Im Falle des USB-Sticks sind mit mehr Datenverlusten zu rechnen, falls der Datenverlust gegen Mittag oder Nachmittag auftritt, da die Speicherung erst am Ende des Tages erfolgt. Aus diesem Grund ist die Datenwiederherstellung mit Git zu bevorzugen.

Die Nachweise für die jeweiligen Datensicherungen finden sie im Anhang unter: TODO(Screenshots in Anhang einfügen)

- USB-Sicherung
- Git-Sicherung

5 Projektmethode

Die verwendete Projektmethode dieser IPA ist SCRUM. Abweichungen und Werkzeuge welcher der Umsetzung dieser IPA nach SCRUM verwendet werden, sind im folgenden Abschnitt beschrieben.

5.1 Github Board

Um die Userstories, Aufwandschätzungen und den Projektstatus zu verfolgen verwende ich Github Projects.

5.1.1 Backlog

Zu Beginn der IPA wurde ein Backlog erstellt indem alle User Stories aufgeführt werden. Die Stories im Backlog müssen noch nicht detailliert spezifiziert sein, sie dienen dazu eine Übersicht über noch offene Aufgaben während der IPA zu erhalten.

5.1.2 Refinement

In der Refinement Spalte werden die Userstories vor dem Sprint Planning detaillierter beschrieben und mit Akzeptanzkriterien versehen. Der Detailbeschrieb dient dazu die Story später im Sprint Planning besser schätzen zu können. Falls eine Userstory zu gross wird, wird sie in dieser Spalte auf zwei oder mehrere Stories unterteilt. Ausserdem werden pro Userstory Akzeptanzkriterien definiert welche erfüllt werden müssen um diese während des Sprints abzuschliessen.

5.1.3 Sprint Backlog

Anfangs Sprint wird immer ein Sprint Planning durchgeführt. Dabei werden die Userstories geschätzt und in den Sprint Backlog gezogen. Am Ende des Sprints sollte der Sprint Backlog leer sein. Ist dies nicht der Fall muss die Story zurück ins Refinement, neu beschrieben werden (falls Änderungen aufgetaucht sind) und muss dann in den nächsten Sprint weitergezogen werden.

5.1.4 In Progress

Während des Sprints werden Ticket in die In Progress-Spalte geschoben sobald die Arbeit daran beginnt.

5.1.5 Done

Eine Userstory kann in die Done-Spalte gezogen werden, wenn alle Akzeptanzkriterien erfüllt wurden. Die Story gilt danach als abgeschlossen.

5.2 Sprints

Die gesamte IPA wird in drei Sprints unterteilt, diese umfassen je eine der folgenden Phasen:

- Initialisierung
- Umsetzung
- Finalisierung

Jedes Ticket wurde mit einem der Phasen gelabeled. So kann abgeschätzt werden, welche Tickets in welchem Sprint erledigt werden müssen.

5.3 Sprint Planning

Das Planning findet immer zu Beginn des nächsten Sprints statt. Während des Sprint Plannings werden die zu erledigenden Stories vom Refinement in den Sprint Backlog geschoben und geschätzt. Um die Planung im Zeitplan besser darzustellen, wird definiert dass die Stories in Stunden anstatt Story Points geschätzt werden. Die niedrigste Schätzung entspricht dabei einem Betrag von 0.5 Stunden.

5.4 Daily

Jeden Morgen findet ein Daily mit der verantwortlichen Fachkraft und der zusätzlichen verantwortlichen Fachkraft statt welche den Stand des Sprintes prüfen und offene Fragen von mir beantworten. Ausserdem präsentiere ich im Daily den Stand der Dokumentation welche meine zuständigen Fachkräfte prüfen und mir Tipps zur Verbesserung geben.

5.5 Verwendungsgrund

Die Projektvorgehensmethod wurde so gewählt, da sie für die IPA mehrere Vorteile bringt:

- **Sprint Ende:** SCRUM zwingt den Entwickler dazu am Ende des Sprintes ein brauchbares Produkt zu haben
- **Agilität:** Wenn eine Story nicht erreicht wurde, kann sie in den nächsten Sprint gezogen werden
- **Daily:** Durch die Dailies wird ein täglicher Austausch zwischen Fachkraft und Kandidat sichergestellt
- **Akzeptanzkriterien:** Mit den Kriterien verhindern wir das abschliessen von halbfertigen Features oder fehlerhafter Software
- **Board:** Durch das Github Projects Board ermöglichen wir eine schnelle Übersicht über den Stand der IPA

6 Projektaufbauorganisation

6.1 Projektrollen in Scrum

Abbildung 6.1: Rollen in Scrum

Rollenbeschreibung	
Product Owner	Der Product Owner vertritt die Interessen des Kunden. Er priorisiert die Aufgaben im Product Backlog
Scrum Master	Der Scrum Master coached die Entwickler und beseitigt Hindernisse. Er sorgt für eine kontinuierliche Verbesserung in der Arbeit.
Entwicklerteam	Das Entwicklerteam arbeitet selbstorganisiert den Sprint Backlog ab. Durch Dailies wird ein laufender Informationsaustausch sichergestellt.

Tabelle 6.1: Rollenbeschreibung

6.2 Projektrollen IPA

Rollenbeschreibung	
Verantwortliche Fachkraft	Unterstützt den Kandidaten von seiten des Lehrbetriebes. Erste Anlaufstelle bei Problemen.
Zusätzliche verantwortliche Fachkraft	Unterstützung für die verantwortliche Fachkraft
Experten	**Validierungsexperte:** Validiert die IPA-Aufgabenstellung. **Hauptexperte:** Verantwortlich für die Bewertung der IPA. **Nebenexperte:** Unterstützung für den Hauptexperten.

Tabelle 6.2: Rollenbeschreibung

6.3 Rollenverteilung

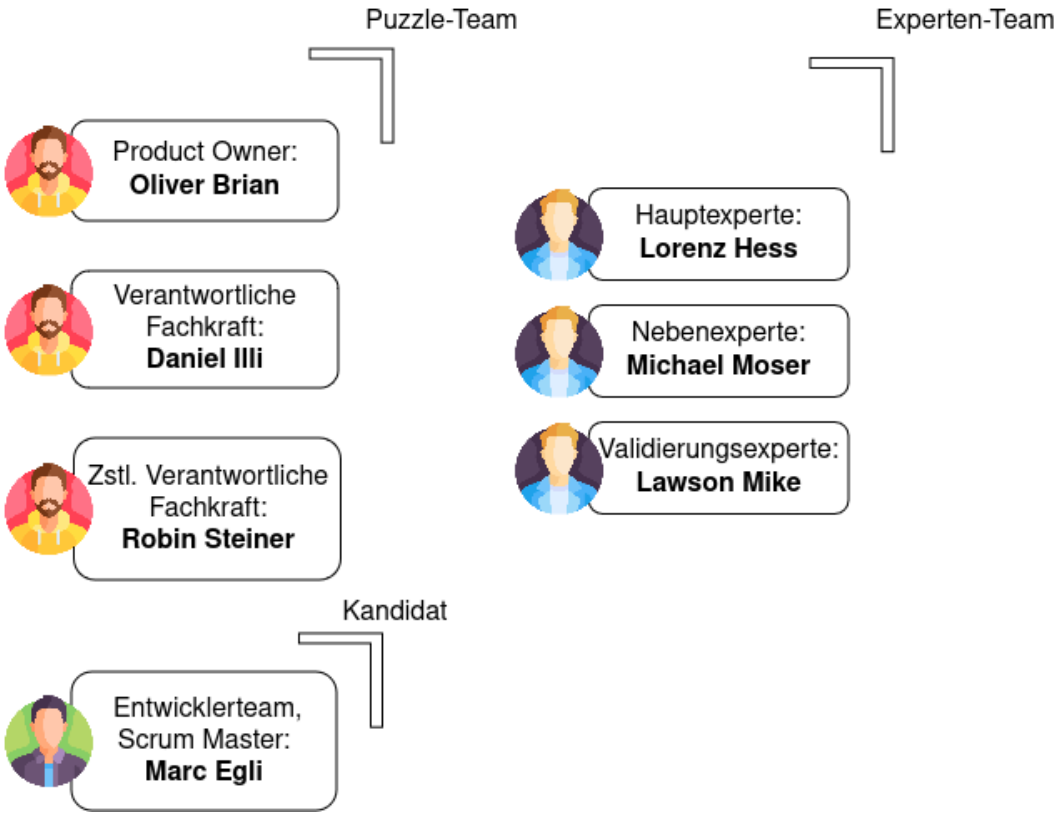


Abbildung 6.2: Rollen in Scrum

Rollenbeschreibung IPA	
Verantwortliche Fachkraft	Daniel Illi
Zusätzliche verantwortliche Fachkraft	Robin Steiner
Validierungsexperte	Lawson Mike
Hauptexperte	Lorenz Hess
Nebenexperte	Michael Moser
Scrum Master	Marc Egli
Development Team	Marc Egli
Kandidat	Marc Egli

Tabelle 6.3: Rollenbeschreibung IPA

7 Zeitplan

7.1 Erläuterung zum Zeitplan

8 Arbeitsjournale

8.1 Tag 1: 14.01.2025

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Planning	Marc Egli	1	1
Zeitplan	Marc Egli	2	2
Aufgabenstellung übernehmen	Marc Egli	1	0.5
Standards aus Github übernehmen	Marc Egli, Nils Rauch	1	1.5
IPA Schutzbedarfanalyse	Marc Egli, Nils Rauch, Olliver Brian, Olliver Dietschi, Thomas Ellenberg	1	0.75
Scrum Beschrieb	Marc Egli	1	1.5
Arbeitsjournal	Marc Egli	0.25	0.5
Backupkonzept	Marc Egli	1	0.25
Total		8.25	8.25

Tabelle 8.1: Tätigkeiten Tag 1

Tagesablauf

Heute bin ich motiviert in die IPA gestartet. Als erstes habe ich am morgen nochmals die Spezifikationen für die Dokumentation, durchgelesen und das Template für die IPA angepasst. Nachdem ich eine passende Struktur hatte, startete ich auch schon direkt mit dem ersten Sprint Planning dieser IPA. Dabei habe ich alle Tasks für den Sprint 1 im Backlog erfasst, diese dann im Refinement detaillierter Beschrieben und am Schluss in den Sprint Backlog geschoben. Die ganze Planung habe ich mit Gihub Projects gemacht, leider kam ich da bezüglich Issues an die Grenzen denn leider kann mann diese nur definieren wenn die Issues einem Projekt, welches NICHT geforked ist,

zugewiesen werden können. Dieses Problem werde ich am Daily morgen mit meiner Fachkraft besprechen, evtl. weis er mehr dazu.

Nach dem Planning begann ich mit dem Bereitstellen des Zeitplans. Ich übernahm das Tempalte welches ich ausgewählt hatte und passte es auf meine drei Sprints in den kommenden zwei Wochen an. Zuerst dachte ich, dass ich den Zeitplan schneller fertigstellen könnte jedoch hatte ich Probleme mit Google Sheets und das anlegen von gemergeden Spalten dauerte lange. Trotzdem ist die Planung aufgegangen und nach 2 Stunden hatte ich einen geeigneten Zeitplan.

Am Nachmittag Startete ich direkt mit dem Dokumentieren, angefangen bei den Standards unserer Firma. Es dauerete länger als gedacht, alle Standards zu sammeln und in die Struktur der Dokumentation zu bringen, weswegen ich dort etwas Zeit verlor. Ein Teil davon konnte ich dann bei der Schutzbedarfsanalyse wieder reinholen. Hier suchte ich den Kontakt mit anderen Mitarbeitern, um herauszufinden wo das Datenschutzkonzept für Hitobito hinterlegt ist. Anscheinend wusste das Niemand aussert Oliver Brian, welcher mir dieses für die Ablage im Anhang zur Verfügung stellte.

Gegen den Ende des Tages habe ich die Projektmethode Scrum Beschrieben und dokumentiert wie ich mich während der IPA organisieren werde. Bezüglich der Aufteilung der Spalten der User Stories bin ich hier noch unsicher, ich werde dies sicher morgen am Daily auch mit Daniel Illi abklären.

Hilfestellungen

- Oliver Brian: Nachfrage Datenschutzkonzept
- Nils Rauch: Nachfrage Sicherheitskonzept / Sicherheitsconventions Puzzle ITC

Reflexion

Ich konnte heute schon einiges dokumentieren und habe nun eine Vorlage von der aus ich einfach weiterarbeiten kann. Zusätzlich habe ich mit Github Project einen Ort an dem ich meinen Fortschritt verwalte und mich selbst organisiere. Probleme gab es nur bei der Beschaffung des Datenschutzkonzeptes und der Arbeit mit Google Sheets.

Was lief gut

Grundsätzlich lief das Dokumentieren selbst sehr gut. Ich konnte alle restlichen Informationen für die Standars oder die Projektmethode schnell Beschaffen und mich dann dem Dokumentieren widmen.

Was lief weniger gut

Weniger gut lief die Arbeit mit Google Sheets und die Arbeit mit der Latex Vorlage. Zum Teil hatte ich recht lange bis ich herausfand wie ich eine Liste anlege oder ein Bild einfügen kann. Ausserdem habe ich mich im Zeitplan verschätzt und heute 9.25 anstatt 8.25 Stunden geschätzt, da ich im Google Sheets einen Fehler gemacht habe. Diesen konnte ich aber schnell korrigieren, so dass ich heute auf geplante 8.25 Stunden komme, welche ich nun auch erreiche.

Meine Erkenntnisse von heute

Mit erweitertem Latex know-how und dem Datenschutzkonzept in den Händen kann ich nun weiter dokumentieren. Ich denke ich werde somit auch weniger Probleme mit Google Sheets und Latex haben, da ich heute schon viele meiner Probleme lösen konnte.

Nächste Schritte

Als nächstes werde ich morgen das Backupkonzept fertig machen und dann direkt zur Projektaufbauorganisation gehen. Nach Abschluss dieser Story kann ich den Sprint 1 Abschliessen und schon in den Sprint 2, der Konzeption / Umsetzung starten.

8.2 Tag 2: 15.01.2024

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Backup Konzept	Marc Egli	0	0.5
Projektaufbauorganisation	Marc Egli	2	2.25
Standards	Marc Egli	0	0.25
Datenschutzkonzept	Marc Egli	0	0.5
Planning	Marc Egli	1	1
Daily	Marc Egli, Daniel Illi	0.25	0.5
Arbeitsjournale	Marc Egli	2	0.25
Total		5.25	5.25

Tabelle 8.2: Tätigkeiten Tag 2

Tagesablauf

Heute konnte ich dank den Erkenntnissen von gestern schnell mit der Latex Dokumentation vorankommen. Das Backup-Konzept konnte ich direkt abschliessen und habe dort sogar noch eine Viertelstunde gespart. Diese brauchte ich wiederum für die Projektaufbauorganisation. Was hier länger gedauert hat, war das erstellen der Diagramme. Ich wollte die Scrum Rollen und Rollenverteilung möglichst übersichtlich machen, was Zeit kostete. Zuletzt waren die Arbeitsjournale geplant, hier habe ich einen Fehler in meiner Planung gemerkt. Ich habe angenommen, dass ich die Arbeitsjournale noch anpassen müsste und wegen der fehlenden Latex-Erfahrung habe ich deswegen zwei Stunden eingeplant.

Allerdings hatten wir schon ein Template für das Arbeitsjournal im Projekt, deswegen hat sich diese Zeit auf 0 Stunden reduziert. Die übrige Zeit habe ich dafür aufgewendet Nachbesserungen an der Dokumentation im Bereich Datenschutzkonzept und Standards zu machen. Zudem hat das Daily auch länger gedauert, welches die nötige Zeit dann rausholen konnte.

Zum Schluss des Tages habe ich den Sprintabschluss und das Planning für die Umsetzungsphase gemacht.

Hilfestellungen

- Nils Rauch: Nachfrage Tool für Erstellung des Zeitplans
- Daniel Illi: Nachfrage Datenschutzkonzept

Reflexion

Heute konnte ich sehr erfolgreich mit der Latex Dokumentation arbeiten. Auch das Planning lief gut, ich denke ich habe nun eine saubere Planung für die Umsetzungsphase welche mir genug Spielraum lässt. Der Fehler mit dem Zeitplan und Arbeitsjournal hat mir zwar Zeit in der Dokumentation gekostet, allerdings ist es besser zu viel Zeit als zu wenig geschätzt zu haben.

Was lief gut

Die Arbeit mit Latex ging heute ohne Probleme voran und meine Effizienz war heute deutlich grösser als gestern.

Was lief weniger gut

Der Fehler im Zeitplan mit den Arbeitsjournalen hat mich in der Planung durcheinandergebracht. Ich habe die Zeiten nun korrekt im Zeitplan vermerkt, damit keine weiteren Probleme darunter entstehen.

Meine Erkenntnisse von heute

Ich sollte vor der Eintragung in den Zeitplan prüfen, ob nicht schon Dokumente existieren welche mir einen Teil der Arbeit abnehmen. Ist dies der Fall, wie bei meinen Arbeitsjournalen kann ich die Aufwandschätzung um ein Wesentliches reduzieren.

Nächste Schritte

Morgen werde ich mit der Umsetzung der IPA starten. Dabei werde ich zuerst die Einführung in das Hitobito Projekt dokumentieren und dann direkt in die Konzeption für eine Filterlösung der Personenlisten und Abos starten. Dies ist ein Schritt der mich zusätzlich motiviert, denn ich kann endlich etwas anderes machen als dokumentieren.

8.3 Tag 3: 16.01.2024

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Einführung	Marc Egli	0.5	0.25
Ist-Zustand	Marc Egli	2	2.5
Soll-Zustand	Marc Egli	0.5	0.25
Persönliche Vorgehensziele	Marc Egli	0.5	0.25
Anforderungen	Marc Egli	1	0.75
Entwurf	Marc Egli, Daniel Illi	3.5	4
Total		8	8

Tabelle 8.3: Tätigkeiten Tag 3

Tagesablauf

Heute war ein sehr anstrengender aber produktiver Tag. Ich konnte zu beginn direkt die Einführung abschliessen. Hier Kommentaren ich schnell durch und sparte etwa eine Viertelstunde. Der Ist-Zustand hat dann länger gedauert. Das lag daran das ich ein Sequenzdiagramm für den Personenlistenfilter und den Abonnementenfilter entworfen habe und danach noch eine möglichst detaillierte Beschreibung der Komponenten lifern wollte. Dafür konnte ich noch vor dem Mittag die Persönlichen Vorgehensziele und Anforderungen definieren. Am Nachmittag startete ich dann mit einem Meeting mit Daniel Illi, meiner Verantwortlichen Fachkraft. Mir waren einige Fragen bezüglich dem Aufbau der Abo-Filter aufgekommen, welche ich mit ihm klären konnte. Nach diesem Meeting konnte ich mit dem erlangten Wissen direkt in den Entwurf starten. Dieser hat mich mental am meisten beansprucht, da ich mich in die Filter-Thematik einarbeiten musste, um nachvollziehen zu können, wie diese aufgebaut sind. Den Entwurf konnte ich nun soweit bringen, dass ich noch die Klassenstruktur der Abonnementenfilter erfassen und die Konzeption für eine Lösung machen muss.

Hilfestellungen

- Daniel Illi: Nachfrage Aufbau Abonnementenfilter

Reflexion

Der heutige Tag war zwar produktiv und ich konnte viele Teile meines Zeitplanes abschliessen, jedoch muss ich unbedingt schneller im Dokumentieren werden. Ich habe morgen noch 2h für den Entwurf und habe das Gefühl das diese Aufwanschätzung sehr knapp wird. Ich werde mich diesbezüglich morgen mit Daniel Illi unterhalten, ob er mir Tipps zur Effizienzsteigerung hat. Meine grösste Angst gilt noch der Umsetzung welche morgen beginnt. Auch wenn ich ein Konzept in Sicht sehe, weiss ich auch das die reine Featureentwicklung wohl meine grösste Schwäche ist, da ich diese bis jetzt selten im Hitobito einsetzen konnte.

Was lief gut

Viele Sektionen in der Dokumentation konnten abgeschlossen werden. Einige davon konnten mit Diagrammen versehen werden, welches die Sektionen schon viel klarer darstellt.

Was lief weniger gut

Weniger gut lief meine Einschätzung der Geschwindigkeit meines Arbeitens. Ich muss schneller werden, ansonst wird es mit der Zeit knapp.

Meine Erkenntnisse von heute

Meine Erkenntniss ist, dass ich schneller im Dokumentieren sein muss, allerdings die Qualität der Dokumentation gleichbehalte.

Nächste Schritte

Morgen werde im Daily mit Daniel Illi besprechen, wie ich meine Effizienz im Dokumentieren steiger kann. Danach werde ich den Entwurf fertigstellen und gegen den späten Morgen mit der Umsetzung meiner Arbeit beginnen. Ziel ist es morgen die Implementation grösstenteils fertigzustellen und ein funktionierendes PoC zu haben.

8.4 Tag 4: 17.01.2024

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Entwurf	Marc Egli, Daniel Illi	2	8
Umsetzung	Marc Egli	6	0
Arbeitsjournal	Marc Egli	0.25	0.25
Total		8.25	8.25

Tabelle 8.4: Tätigkeiten Tag 4

Tagesablauf

Heute wollte ich nach Planung den Entwurf möglichst schnell abschliessen, jedoch war das Gegenteil der Fall. Ich musste mich noch mehr in die Abläufe des Programmes momentan arbeiten und das Erstellen des Variantenentscheids dauerte ewig. Zuletzt hat auch die Ausarbeitung enorm lange gebraucht. Das heisst ich muss am Dienstag umso mehr Gas geben um die Umsetzung abzuschliessen. Damit ich mehr Zeit für diese gewinne, werde ich einen halben Tag der Finalisierung umbuchen und für die Umsetzung verwenden, wenn ich merke, dass es nicht mehr reicht. Dies wird planmässig am Dienstag im Planning geschehen.

Hilfestellungen

- Daniel Illi: Nachfrage Aufbau des Konzeptes

Reflexion

Obwohl ich sehr viel Zeit für den Entwurf gebrauch habe, bin ich dennoch froh, dass ich nun einen konkreten Plan für die Umsetzung habe. So kann ich direkt am Dienstag mit dieser beginnen. Ich hätte vielleicht ein paar Teile des Entwurfes weniger detailliert machen müssen, auf der anderen Seite denke ich mir aber, dass ich bei der Umsetzung um genau diese Definition der Details sehr dankbar sein werde.

Was lief gut

Nach der Dokumentation des Entwurfs verstehe ich nun genau wie der Abonnementenfilter und Personenlistenfilter aufgebaut sind. Ich habe ein Konzept welches eine Brücke zu den beiden Filterungsprozessen bildet und die alte Funktionalität immer noch gewährleistet, was mich motiviert dieses schon bald umzusetzen.

Was lief weniger gut

Trotz der guten Dokumentation ging wieder enorm viel Zeit verloren. Hier hätte ich mehr Zeit im Planning schätzen müssen und mindestens gleich viel Zeitplan für den Entwurf wie die Umsetzung einrechnen müssen.

Meine Erkenntnisse von heute

Meine Erkenntnisse liegen darin das ich weiss wie ich mein IPA Feature umsetzen will und das ich das nächste Mal im Planning wesentlich mehr Zeit für den Entwurf einplane.

Nächste Schritte

Meine nächsten Schritte sind das beginnen mit der Umsetzung meines definierten Konzepts.

8.5 Tag 5: TODO: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
TODO: Tätigkeit	TODO: Beteiligte Personen	TODO: Stunden Soll	TODO: Stunden Ist
Total		TODO: Stunden Soll Total	TODO: Stunden Ist Total

Tabelle 8.5: Tätigkeiten Tag 5

Tagesablauf

Hilfestellungen

- TODO: Hilfestellungen auflisten

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.6 Tag 6: TODO: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
TODO: Tätigkeit	TODO: Beteiligte Personen	TODO: Stunden Soll	TODO: Stunden Ist
Total		TODO: Stunden Soll Total	TODO: Stunden Ist Total

Tabelle 8.6: Tätigkeiten Tag 6

Tagesablauf

Hilfestellungen

- TODO: Hilfestellungen auflisten

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.7 Tag 7: TODO: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
TODO: Tätigkeit	TODO: Beteiligte Personen	TODO: Stunden Soll	TODO: Stunden Ist
Total		TODO: Stunden Soll Total	TODO: Stunden Ist Total

Tabelle 8.7: Tätigkeiten Tag 7

Tagesablauf

Hilfestellungen

- TODO: Hilfestellungen auflisten

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.8 Tag 8: TODO: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
TODO: Tätigkeit	TODO: Beteiligte Personen	TODO: Stunden Soll	TODO: Stunden Ist
Total		TODO: Stunden Soll Total	TODO: Stunden Ist Total

Tabelle 8.8: Tätigkeiten Tag 8

Tagesablauf

Hilfestellungen

- TODO: Hilfestellungen auflisten

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.9 Tag 9: TODO: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
TODO: Tätigkeit	TODO: Beteiligte Personen	TODO: Stunden Soll	TODO: Stunden Ist
Total		TODO: Stunden Soll Total	TODO: Stunden Ist Total

Tabelle 8.9: Tätigkeiten Tag 9

Tagesablauf

Hilfestellungen

- TODO: Hilfestellungen auflisten

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.10 Tag 10: TODO: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
TODO: Tätigkeit	TODO: Beteiligte Personen	TODO: Stunden Soll	TODO: Stunden Ist
Total		TODO: Stunden Soll Total	TODO: Stunden Ist Total

Tabelle 8.10: Tätigkeiten Tag 10

Tagesablauf

Hilfestellungen

- TODO: Hilfestellungen auflisten

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

9 Persönliches Fazit

Teil II

Projektdokumentation

Hitobito: Neue Generation von Personen-Filtern
Autor: Marc Egli

10 Einführung

Puzzle ITC ist ein schweizer Anbieter für Softwarelösungen. Die Firma hat ihren Hauptsitz in Bern, besitzt aber weitere Standorte in Zürich, Luzern und Deutschland (Thüringen). Puzzle bietet als Unternehmen die ganze Palette an IT-Services an, von Digital Transformation bis hin zu Data Analytics. Nebst den vielen Angeboten tritt Puzzle dabei immer seine Grundwerte nach aussen, welche im Puzzlehouse abgebildet werden.

Abbildung 10.1: Rollen in Scrum

Hitobito ist eines der Angebote von Puzzle. Es ist ein Community-Management Tool und als Open-Source Projekt auf Github zu finden. Das Tool wird von zahlreichen Verbänden, Parteien und Organisationen verwendet und befindet sich darum in einer kontinuerilichen Weiterentwicklung. Mit dem Wagons-Gem ermöglicht es Hitobito zudem spezielle Kundenanpassungen in einem eigenen ”Wagonßu vollziehen, ohne die Software anderer Kunden mit-anzupassen.

Ich selbst arbeite jetzt seit einem halben Jahr im Hitobito und nahm darin vor allem Upgrades und Migrationen vor. So durfte ich bspw. das Upgrade von RoR (Ruby on Rails) von 6.1 auf 7.1 vornehmen oder die Migration von MySQL auf Postgres vollziehen.

Da Hitobito von zahlreichen Kunden verwendet wird, ist die Applikation über die Jahre gewachsen. Viele Features wurden implementiert, um sie schnell dem Kunden zur Verfügung zu stellen. Mit einem immer wachsenden Anforderungskatalog ergaben sich dadurch komplexe Arbeitsabläufe welche im Tool etabliert wurden. Einer dieser komplexen Abläufe ist die Filterung nach Personen oder Abonnemente.

Mit dieser IPA soll die Filterung zwischen diesen zwei Entitäten homogenisiert werden. Um dies zu tun, sollen zuerst zwei bis drei Konzepte ausgearbeitet und anschliessend in einem Variantenentscheid evaluiert werden. Für die Lösungsvariante wird in einem weiteren Schritt ein PoC (Proove of Concept) implementiert.

Nach der IPA soll basierend auf der neuen Filterlogik ein neues UI entworfen werden, um nebst der Ordnung im Backend eine besser User Experience für den Benutzer zu schaffen.

In einer Zeit in welcher Unternehmen mehr den je Wert auf ein sauberes Design und der User Experience von Webseiten und Applikationen geben, das auch in einer älteren Applikation zu etablieren. Gerade bei einem Community-Management Tool wie Hitobito, welches tagtäglich von Personen bedient werden, welche nicht das technische Know-How dahinter besitzen, ist es wichtig Arbeitsabläufe so einfach wie möglich zu entwerfen, um maximale Effizienz für diese Personen zu garantieren. Durch eine Vereinfachung der Hitobito-Fitler machen wir damit einen ersten Schritt in die richtige Richtung.

11 Analyse

In der Analyse der IPA wird der Rahmen geschaffen in welchem man später während des Implementierens arbeitet. Sie befasst sich mit der Aufnahme von Ist- und Zielzustand und definierte Funktionale sowie nicht funktionale Anforderungen. Es wird definiert wo sich die IPA abgrenzt.

11.1 Ist-Zustand

11.1.1 Personenlisten

Aktuell kann ein Nutzer über den Button Neuer-Filter auf die Filter Seite navigieren.

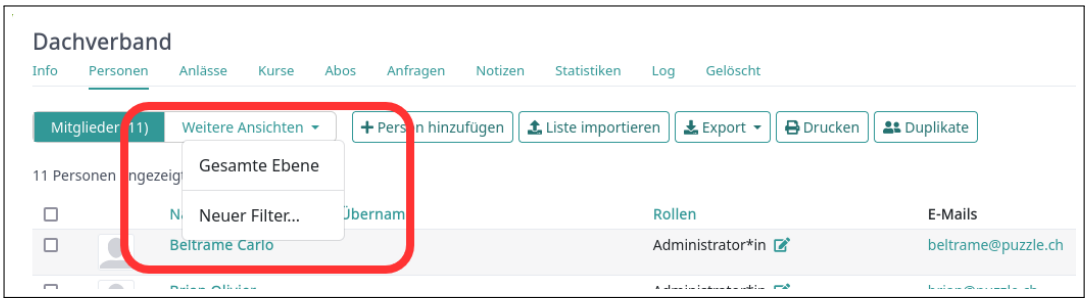


Abbildung 11.1: Hitobito Personenlisten Filtererstellung

Auf dieser definiert er die Filterungskriterien für die Attribute Rollen, Qualifikationen, Felder, Sprache und Tags.

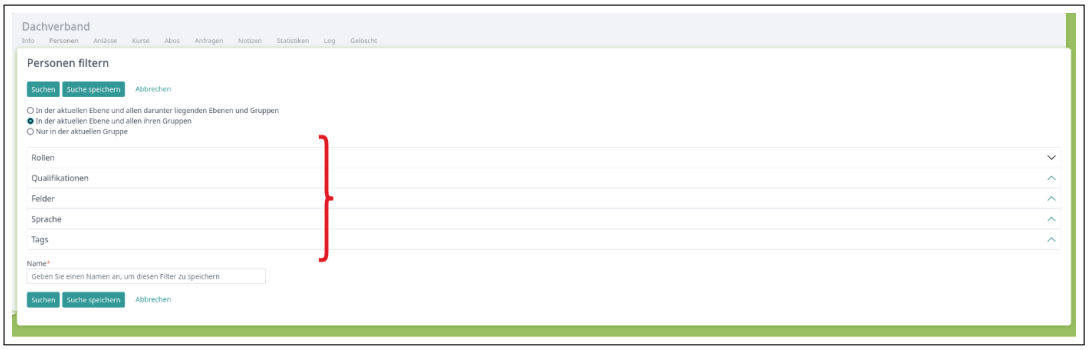


Abbildung 11.2: Hitobito Personenlisten Filterkriterien

Anschliessend ist es dem Nutzer möglich seinen Filter über einen Button für die Wiederverwendung zu speichern.

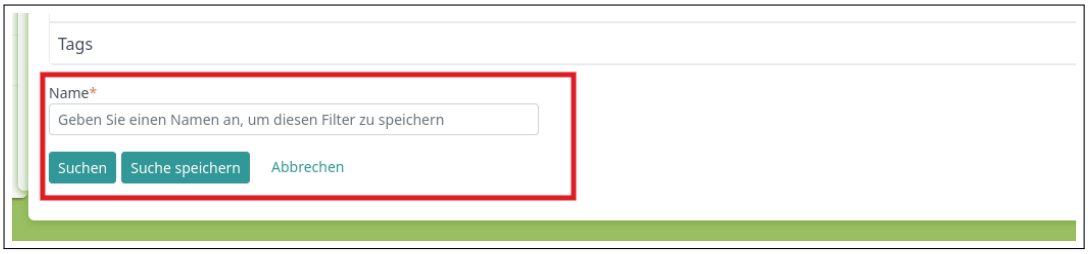


Abbildung 11.3: Hitobito Personenlistenfilter Speicherung

Technisch sind die Personenlisten-Filter nach folgendem Sequenzdiagramm aufgebaut:

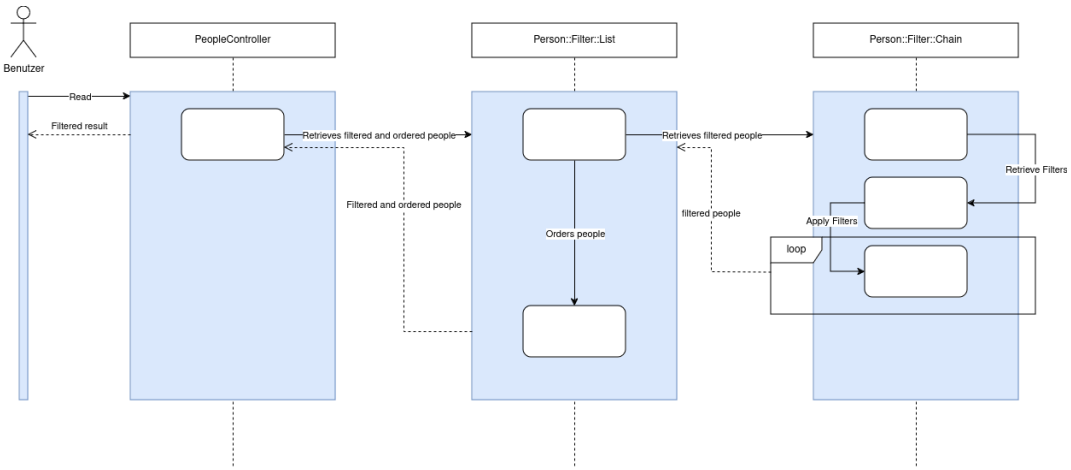


Abbildung 11.4: Sequenzdiagramm Personenlisten Filter

Beschreibung Sequenzdiagramm	
PeopleController	Der PeopleController nimmt den Request des Benutzers entgegen und erwidert die gefilterten und sortierten Personen auf welche der Benutzer zugreifen darf.
Person::Filter::List	Personen auf welche der Benutzer keinen Zugriff hat werden rausgefiltert und anschliessend sortiert.
Person::Filter::Chain	Definiert anhand der Request Parameter vom Controller die Filter und wendet diese in einem Loop via chain-pattern an.

Tabelle 11.1: Beschreibung Sequenzdiagramm

11.1.2 Abonnemente

Auf den Abonnementen kann der Nutzer Filterkriterien in den globalen Bedingungen definieren.

Abbildung 11.5: Hitobito Globale Bedingungen

Unter diesen kann der Nutzer per Dropdown entscheiden für welche Attribute er die Filterkriterien definieren möchte. Er kann auch bereits gesetzte Kriterien entfernen. Pro Filterkriterium entscheidet er im weiteren, mit welcher Genauigkeit nach diesem Filterkriterium gesucht wird. Bei Zahlen ist es möglich die Genauigkeiten ist genau, ist höher als und ist kleiner als einzustellen. Bei Textvergleichen sind es ist genau, enthält, enthält nicht.

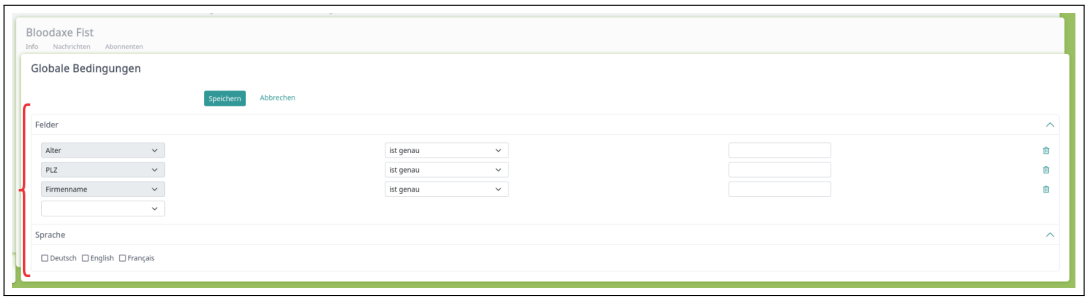


Abbildung 11.6: Hitobito Filterkriterien

Hat der Nutzer seine Filterkriterien und die dazugehörigen Genauigkeiten definiert, kann er sie über den Speicher-Button persistieren. Im Anschluss werden die ausgewählten Filterkriterien in den Globalen Bedingungen angezeigt und ein Success-Alert ausgelöst der die Aktualisierung bestätigt.

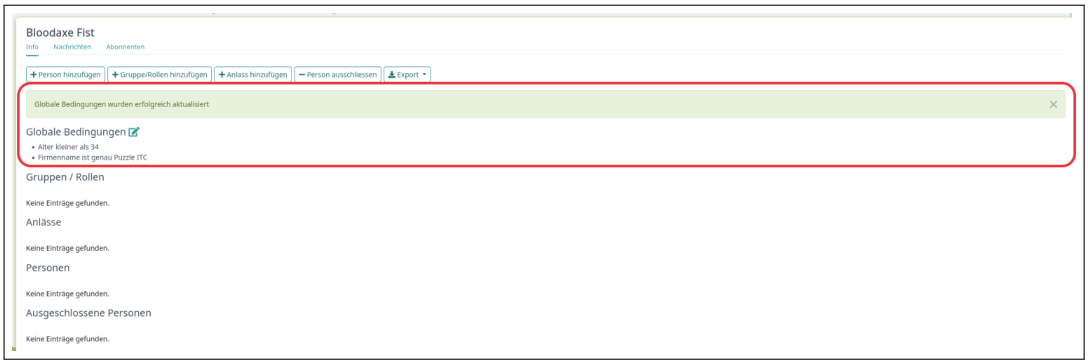


Abbildung 11.7: Hitobito Filterkriterien

Technisch haben sind die Abonnemente nach folgendem ERD aufgebaut.

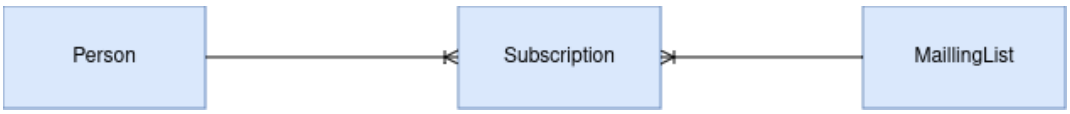


Abbildung 11.8: Hitobito Subscription ERD

Es ist zu beachten: Eine Person kann mehrere Subscriptions besitzen und jede Subscriptions ist einer Mailling list zugerodnet. Somit muss bei der Filterung nach Subscriptions zuerst die MaillingList included werden, damit man auf dieser danach die Filterung ausführen kann.

Die Globalen Filterungskriterien für die Abonnemente werden in dem Modell der MaillingList abgespeichert. Dadurch ergibt sich folgendes Sequenzdiagramm.

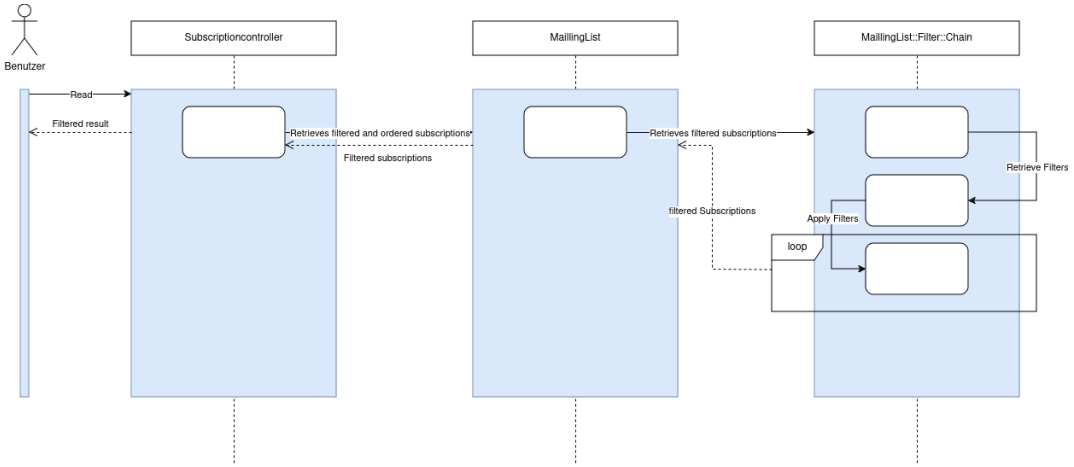


Abbildung 11.9: Hitobito Abonnementen Sequenzdiagramm

Beschreibung Sequenzdiagramm	
SubscriptionController	Der SubscriptionController nimmt den Request des Benutzers entgegen und erwidert die gefilterten und sortierten Personen auf welche der Benutzer zugreifen darf.
MaillingList	Fetcht Subscriptions aus der Datenbank und macht Aufruf zum MaillingList::Filter::Chain
MaillingList::Filter::Chain	Holt die Filterkriterien aus der Datenbank und filtered die Subscriptions danach aus.

Tabelle 11.2: Beschreibung Sequenzdiagramm

11.2 Soll-Zustand

Das neue Konzept für die Abonnementen- und Personenlistenfilterung soll durch den gleichen Prozess laufen. Die beiden Filterungsprozesse sollen dabei zu einem homogenisiert werden und trotzdem die gleichen Funktionalitäten bieten. Bestehende Datenmodelle sollen ebenfalls fusioniert werden.

11.3 Persönliche Vorgehensziele

Zeitrahmen	
Beschreibung	Der erstellte Zeitplan, Meilensteine und Sprints werden erfolgreich eingehalten.
Messbarkeit	Die IPA wird komplett und in der definierten Frist abgeben.

Tabelle 11.3: Zeitrahmen

Filterprozesse	
Beschreibung	Die Verständnis der Hitobito Filterlogik wird vertieft.
Messbarkeit	Zukünftig dient Kandidate als Anlaufstelle für Fragen zur Filterung und kann dieses Wissen in anderen Aufträgen anwenden.

Tabelle 11.4: Filterprozesse

Ruby on Rails	
Beschreibung	Das Wissen rund um das Ruby on Rails Framework und Aufbau von Domain und Modellklassen wird vertieft.
Messbarkeit	Erlangtes Wissen kann in zukünftiger Entwicklung an Features eingesetzt werden.

Tabelle 11.5: Ruby on Rails

Konzeption	
Beschreibung	Der Kandidat ist in der Lage Features oder Umstrukturierungen an einer Applikation selbständig zu konzipieren.
Messbarkeit	Konzept für neues Filterverhalten wurde sauber aufgestell und Entscheidung dafür begründet.

Tabelle 11.6: Konzeption

11.4 Anforderungen

11.4.1 Nicht funktionale Anforderungen

Erweiterbarkeit, NfA. 1	
Beschreibung	Bei der Implementation der neuen Filterlogik wird beachtet, dass zukünftig weitere Filterkriterien oder Genauigkeiten vom Kunden gewünscht werden können.
Messbarkeit	Die Implementation wurde nachhaltig umgesetzt und bietet Möglichkeiten zur Erweiterung.

Tabelle 11.7: Erweiterbarkeit

Performance, NfA. 2	
Beschreibung	Die Implementation des neuen Filterungskonzeptes soll die gleiche Leistungsfähigkeit wie die jetztige Implementation vorweisen.
Messbarkeit	Ladezeit der Personenlisten und Abonnemente bleibt gleich.

Tabelle 11.8: Performance

Security, NfA. 3	
Beschreibung	Filterungen sollen auf dem Datensatz gemacht werden, welche durch das can-can-can Gem verfiziert hat.
Messbarkeit	Nutzer können keine Daten einsehen, auf welche sie keine Berechtigungen haben.

Tabelle 11.9: Security

DRY, NfA. 4	
Beschreibung	Der Code wurde nach dem DRY (Don't Repeat Yourself) Prinzip implementiert.
Messbarkeit	Es finden sich keine doppelten Klasse oder unnötige Wrapper.

Tabelle 11.10: DRY

Dokumentation, NfA. 5	
Beschreibung	Code wurde dokumentiert.
Messbarkeit	Codingfiles sind mit verständlichen Kommentaren versehen.

Tabelle 11.11: Dokumentation

11.4.2 Funktionale Anforderungen

Konzept, fA. 1	
Beschreibung	Das erstellte Konzept führt beide Filterprozesse zusammen.
Messbarkeit	Es gibt im Hitobito ein homogenisierter Fitlerprozess der für weiter Filterungen wiederverwendet werden kann.

Tabelle 11.12: Konzept

Filterkriterien, fA. 2	
Beschreibung	Alle bisherigen Filterungskriterien werden weiterhin unterstützt.
Messbarkeit	UI ist immer noch gleich bedienbar wie vor der IPA.

Tabelle 11.13: Filterkriterien, fA. 2

11.5 Abgrenzung

Während der IPA wird auf einem geforkten Repository gearbeitet, sowohl im Hitobito Core-Wagen wie auch im Generic Wagon. Jeglich Commits und Push erfolgen auf den Master Branch des geforkten Repositories. Durch diese Abkapselung wird sichergestellt das die Weiterentwicklung von Hitobito nicht den Erfolg dieser IPA gefährden.

12 Entwurf

12.1 Anwendungskonzept

Das Anwendungskonzept beschreibt wie ein Benutzer die Funktionalität dieser Arbeit verwendet und welche Anwendungsfälle daraus entstehen.

12.1.1 Anwendungsdiagram

Es bilden sich 4 konkrete Use-Cases:

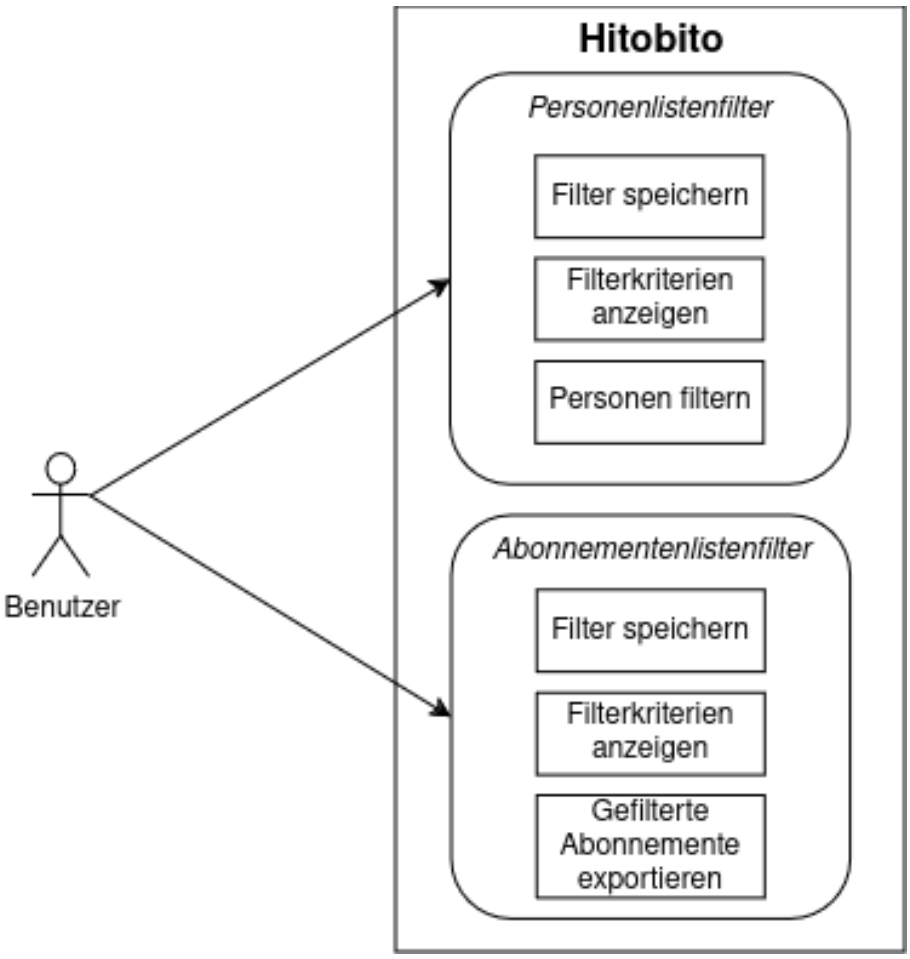


Abbildung 12.1: Hitobito Filter Use Case Diagramm

12.1.2 Anwendungsfälle

Aus dem Anwendungsdiagram werden die 4 Use-Cases entnommen und hier im Detail beschrieben.

Filter speichern	
Kurzbeschreibung	Der Benutzer kann die ausgewählten Fitlerkriterien speichern.
Vorbedingungen	• Der Benutzer besitzt die nötigen Rechte um eine Filter zu erstellen • Mind. ein Filterkriterium wurde ausgewählt
Ablauf	1. Benutzer benennt den Filter, optional und nur bei Personenliste 2. Benutzer klickt auf speichern
Resultat	Der Filter wurde in der Datenbank persistiert und ein Success-Alert wird ausgegeben.

Tabelle 12.1: Anwendungsfall: Filter speichern

Filterkriterien anzeigen	
Kurzbeschreibung	Der Benutzer kann die Filterkriterien der gespeicherten Filter einsehen.
Vorbedingungen	• Der Benutzer hat einen Filter gespeichert
Ablauf	1. Navigiert zum Filter
Resultat	Die Filterkriterien werden dem Benutzer angezeigt.

Tabelle 12.2: Anwendungsfall: Filterkriterien anzeigen

Personen filtern	
Kurzbeschreibung	Der Benutzer kann die definierten Personenlistenfilter auf eine Liste von Personen anwenden.
Vorbedingungen	• Benutzer besitzt Rechte um auf eine Personenliste zuzugreifen • Benutzer hat einen Personenlistenfilter für diese Liste gespeichert
Ablauf	1. Benutzer Navigiert zum Personenlistenfilter 2. Benutzer klickt auf Filternamen
Resultat	Die Personen in der Personenliste werden gefiltert und dem Benutzer angezeigt.

Tabelle 12.3: Anwendungsfall: Personen filtern

Gefilterte Abonnemente exportieren	
Kurzbeschreibung	Der Benutzer kann die Abonnemente als PDF, CSV, Excel, etc. exportieren, dabei werden die gesetzten Abonnementenfilter auf diesen Export angewendet.
Vorbedingungen	• Benutzer hat Rechte um Abonnemente zu bearbeiten • Benutzer hat einen Abonnementenfilter gespeichert
Ablauf	1. Benutzer zum Abonnementenfilter 2. Benutzer wählt Zieldatei aus Dropdown aus 3. Benutzer exportiert Abonnementenliste
Resultat	Die Personen in der Personenliste werden gefiltert und dem Benutzer angezeigt.

Tabelle 12.4: Anwendungsfall: Gefilterte Abonnemente exportieren

12.2 Systemkonzept

Bei dieser Arbeit wird mit einem bestehenden System gearbeitet, dieses muss entsprechend angepasst werden. Um die nötigen Anpassungen besser sichtbar zu machen, wird im folgenden Abschnitt gezeigt, welche Services von den Anpassungen betroffen, was der aktuelle Stand der Logik und was mögliche Konzepte für die Erweiterung sind. Danach wird aufgrund eines Variantenentscheids eine Lösungsvariante ausgearbeitet und als Plan für das PoC definiert.

Abbildung 12.2: Services

12.2.1 Betroffene Services

Hitobito wird hauptsächlich in zwei Services unterteilt, der Rails Applikation und der Postgres Datenbank.

Rails Applikation / Webserver

Die Rails Applikation verwaltet die Business Logik von Hitobito. Die Änderungen / Erweiterugnen dieser Arbeit werden alle in diesem Service vorgenommen. Je nach Kunde werden hier Code Teile aus den definierten Wagons verwendet. In dieser Arbeit wird jedoch ausschliesslich der Core und der Generic Wagon angepasst.

Postgres Datenbank

Die Datenbank von Hitobito läuft auf Postgres. Sämtliche Abfragen auf die Postgres Datenbank werden via SQL-Queries gemacht. Active Records is das ORM (Object Relational Mapping) welches verwendet wird.

12.2.2 Status quo

Personlistfilter

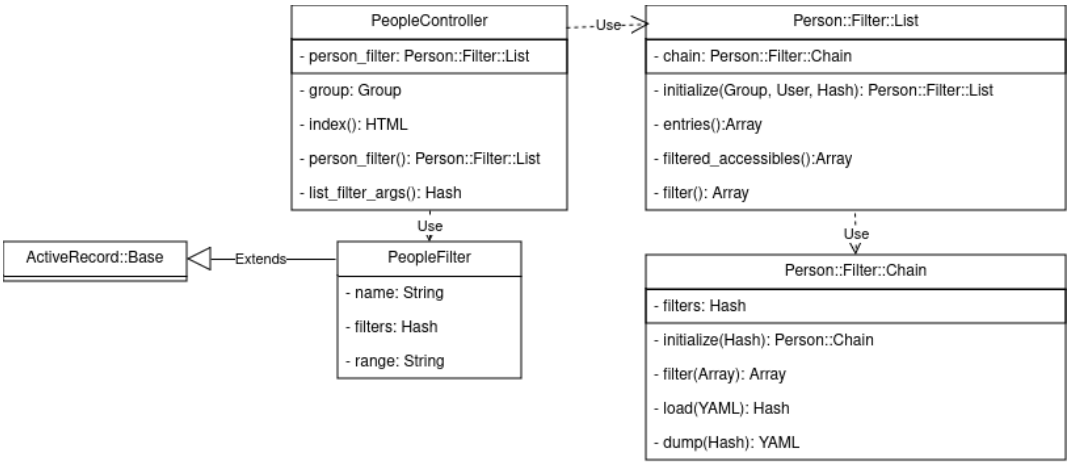


Abbildung 12.3: Klassenstruktur Personenlistenfilter

In der Abbildung oben ist die Klassenstruktur der Personenlistenfilterlogik zu sehen. Wenn der Benutzer einen Filter-Request absetzt trifft dieser erstmals auf die `index` Methode im PeopleController. In dieser ruft der Controller klassenintern die Methode `person_filter` auf. Darin wird eine neue Instanz der `Person Filter List` erstellt und die Filterparameter übergeben. Die Parameter werden über die Methode `list_filter_args` geholt. Wurde schon ein Fitler abgespeichert, holt sich die Methode die persistierte Instanz der `PeopleFilter` Klasse und zieht die Parameter aus dem Attribut `filters`. Ist noch kein Filter definiert übergibt die Methode die Parameter des Requests der Instanz.

Die `Person Filter List` sorgt mit `filtered_acessibles` dafür das nur Daten zurückkommen, auf welche der Benutzer Zugriff hat. Die Methode `filter` ruft intern eine Instanz von `Person Filter Chain` auf und gibt mit durch die Methode `filter` die gefilterten Daten zurück.

Um diese Chain später in dem PeopleFilter model zu speichern, überschreibt die Klasse `Person Filter Chain` die `load` und `dump` Methode von Active Rails, um später die Filter Parameter selbständig in einen Hash zu transformieren.

Abonnementenfilter

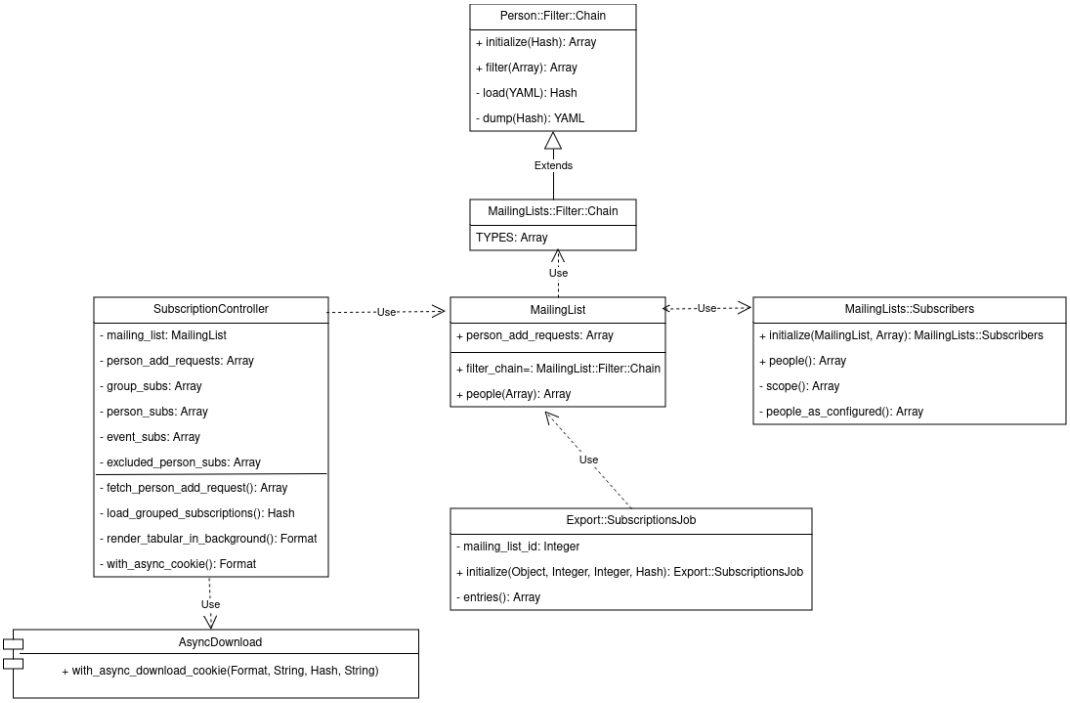


Abbildung 12.4: Klassenstruktur Personenlistenfilter

In der Klassenstruktur des Abonnementenfilters trifft ein Request des Benutzers zuerst auf die `index` Action im SubscriptionController. Danach wird `fetch_person_add_request` aufgerufen welche auf die MailingListe zugreift um die `add_requests` zu fetchen. Danach wird durch die Methode `load_grouped_subscriptions` die einzelnen subscribers geladen. Diese werden dann in der Filterungsansicht angezeigt.

Die Filterung selbst wird nur beim Export von Abonnementen gemacht. Wenn der Benutzer ein CSV der Abonnemente herunterlädt, trifft der Request wieder auf die `index` Action im SubscriptionController. Als nächstes wird durch die Methode `render_tabular_in_background` ein weiterer Aufruf zur `with_async_download_cookie` Methode gemacht, welche sich im Module `AsyncDownload` befindet. Diese Methdoe lädt das CSV herunter. Die Daten dafür bekommt die Methode vom `Export SubscriptionsJob` welcher mit der Methode `entries` die MailingListe aufruft und darin durch die Methode `people` eine Instanz der Klasse `MailingLists::Subscribers` erstellt. Die Klasse holt sich durch `people_as_configured` die gefilterten Abonnemente. Die Filterung geschieht durch einen Call zurück zur Mailingliste welche dann durch die `filter_chain` Methode die Klasse `MailingLists::Filter::Chain` aufruft. Die Filterung funktioniert von hier an ähnlich wie bei den Personenlisten, da `MailingLists::Filter::Chain` von Person::Filter::Chain

erbt. Mit den gefilterten Abonnementen wird `scope` aufgerufen welches schlussendlich doppelte Abonnemente entfernt und zurückgibt. Anhand der gefetchten Daten kann der CSV Export vom `AsyncDownload` Modul abgeschlossen werden.

12.2.3 Lösungsvarianten

In diesem Abschnitt werden Lösungsvariantent für diese Arbeit definiert. Es sind Grobkonzepte welche genug Informationen darstellen, um eine Evaluation jeder Variante durchzuführen.

Lösungsvariante 1

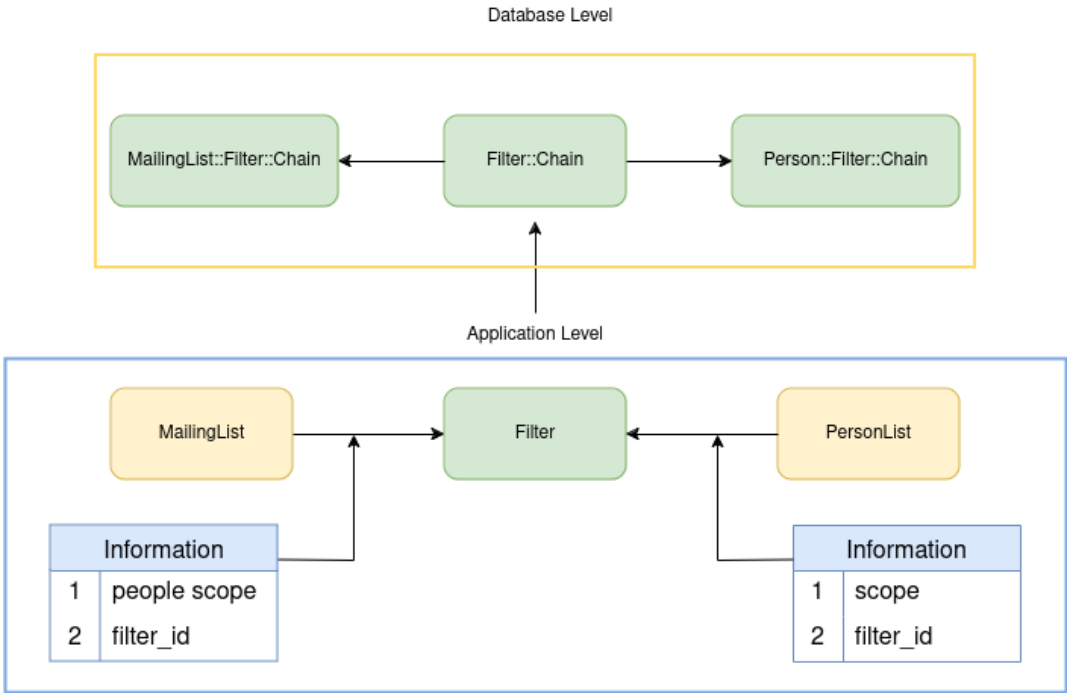


Abbildung 12.5: Neues Filterkonzept 1

Die Idee dieser Variante ist, dass die Filterung selbst nur noch über eine Klasse und ein Modell gelöst wird. Die Klasse Filter soll dabei die nötigen Parameter entgegennehmen und die neue Entität Filter Chain aufrufen. Hier handelt es sich um das Parent Modell der MaillingList Filter Chain und der Person Filter Chain. Diese Beziehung könnte auf Datenbanklevel mit einer Single Table Inheritance Strategie realisiert werden. Ziele dieser Lösungsvariante ist den Prozess der Filterung auf eine Klasse zu reduzieren.

Lösungsvariante 2

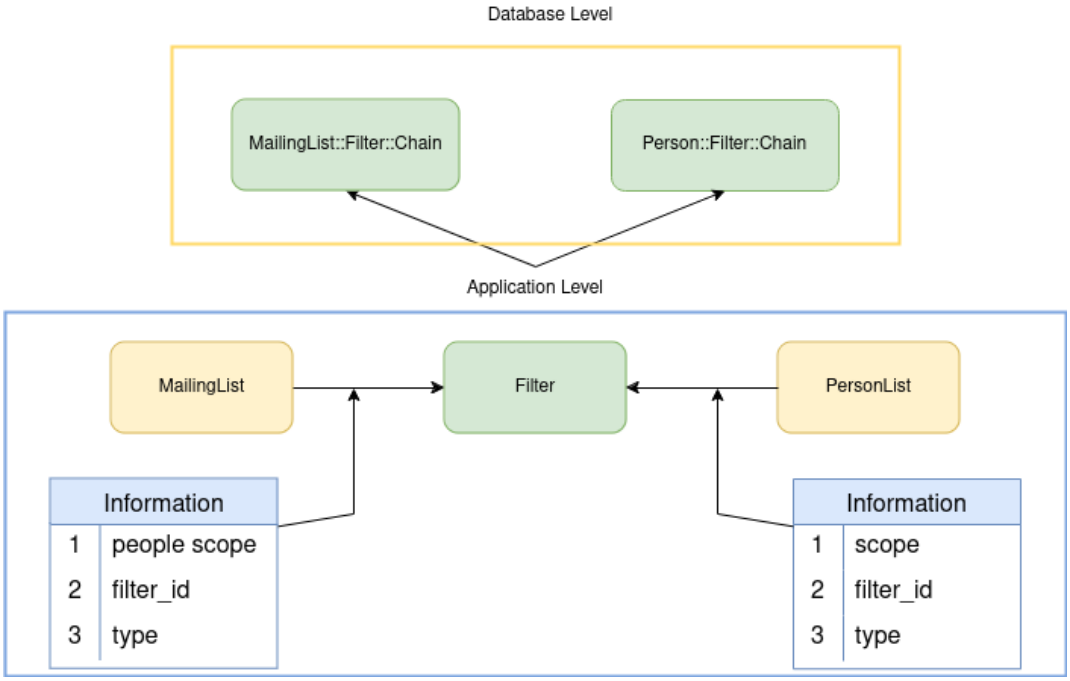


Abbildung 12.6: Neues Filterkonzept 2

Die Lösungvariante 2 ist ähnlich aufgebaut wie die erste. Allerdings wird in dieser Varainte auf die Single Table Inheritance Strategie verzichtet. Stattdessen soll mit eine Attribute ”typeëntschieden werden von welchem Typ die Anfrage für eine Filterung kommt. Die Filterklasse entscheidet dann auf welchen Table sie die SQL-Abfrage richten muss und fetcht die Daten.

Lösungsvariante 3

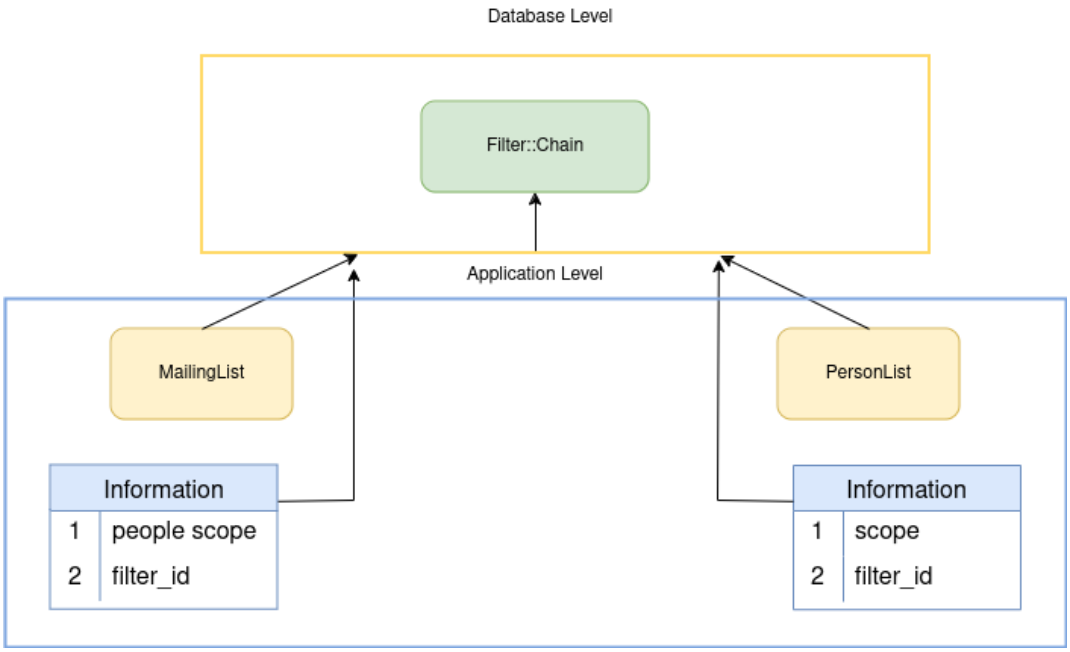


Abbildung 12.7: Neues Filterkonzept 3

Diese Variante lässt den Applicationlayer bestehen wie bisher, es werden keinen neuen Klassen implementiert. Allerdings greifen die MailingList und die PersonList nun direkt auf ein neues Model zu, der Filter Chain. Diese beinhaltet die einen Zusammenzug der Filterdaten die gespeichert wurden. Dazu muss das `filter_chain` Attribut und die People Filter Entitäten in diesen Table migriert werden. Die Information über den Typ nach welchem gefiltert wird, ist nicht mehr nötig da die Filterkriterien direkt mit der Filter ID vom Table geholt werden können.

12.2.4 Variantenentscheid

Um eine geeignete Entscheidung für eine der beschriebenen Lösungsvariantetn zu treffen, wird eine Bewertungsmatrix verwendet welche jede Lösungsvarainte anhand definierter Kriterien beurteilt. Die Kriterien können mit Punkten von 1 bis 10 bewertet werden, wobei 1 Punkt für das Nicht-Erfüllen eines Kriteriums und 10 für das Erfüllen des Kriteriums steht.

Folgende Kriterien wurden definiert:

Kriterium	Beschreibung	1 Punkt	10 Punkte
Zeitaufwand	Wie viel Zeit wird für die Implementation benötigt?	Grosser Zeitaufwand, IPA ist mit diesem Konzept nicht umsetzbar	Kleiner Zeitaufwand, IPA ist problemlos umsetzbar.
Einführung	Ist das Konzept einfach in die produktive Umgebung einzuführen? Müssen Migrationen vorgenommen werden?	Einführung in produktive Umgebung ist unmöglich	Einführung in produktive Umgebung ist problemlos möglich.
Wissen	Kann der Kandidate mit der Umsetzung des Konzeptes etwas lernen?	Kandidat lernt nichts neues dazu	Es kann neues Wissen erarbeitet werden
Performance	Ist das Konzept performant? Spart das Konzept Zeit / Requests?	Konzept ist nicht performant, Filterungen werden durch Implementation deutlich langsamer	Hohe Performance, es kann viel Zeit durch die Implementation des Konzepts gespart werden.

Tabelle 12.5: Variantenentscheid Kriterien

Bewertungen

Im folgenden Abschnitt werden den definierten Kriterien Gewichtungen hinzugeführt, da nicht jedes Kriterium gleich wichtig für diese IPA ist.

Gewichtungen in %

- **Zeitaufwand 40%:** Um eine funktionelle Lösung am Ende der IPA aufweisen zu können, wird dem Zeitaufwand eine hohe Gewichtung zugerodnet
- **Einführung 15%:** Es ist wichtig eine Lösung zu implementieren welche schnell ihren Weg in die produktive Umgebung findet, da die Einführung von der IPA ausgenommen ist, wird diesem Kriterium eine geringere Gewichtung zugeordnet.
- **Wissen 30%:** Damit von der IPA profitiert werden kann, sollte von der Kandidat stets eine Wissenserweiterung durch deren Umsetzung erlangen.
- **Performance 15%:** Ist eine Applikation zu langsam und benötigt mehrere Minute bis sie Resultate geladen hat, kann das den Benutzer schnell vor den Kopf stossen und dafür führen das dieser die Applikation in Zukunft nicht mehr verwendet.

Lösungsvariante 1

Kriterium	Bewertung	Beschreibung
Zeitaufwand	2	Grösster Zeitaufwand, Konzeption mit den meisten Neuimplementationen.
Einführung	4	Einführung möglich, es sollten keine grossen Schwierigkeiten auftreten
Wissen	10	Durch viele Neuimplementationen kann hier am meisten profitiert werden.
Performance	5	Performance kann mit Sicherheit beibehalten werden.

Tabelle 12.6: Bewertung Lösungsvariante 1

Lösungsvariante 2

Kriterium	Bewertung	Beschreibung
Zeitaufwand	4	Mittlerer Zeitaufwand, eine Neuimplementation
Einführung	5	Einführung problemlos möglich, es werden keine Migrationen benötigt
Wissen	8	Durch die Neuimplementation einer Filterklasse kann profitiert werden, Punktabzug aufgrund der nicht vorhandenen Umsetzung der Single Table Inheritance
Performance	5	Performance kann mit Sicherheit beibehalten werden.

Tabelle 12.7: Bewertung Lösungsvariante 2

Lösungsvariante 3

Kriterium	Bewertung	Beschreibung
Zeitaufwand	4	Mittlerer Zeitaufwand, keine Neuimplementationen jedoch müssen Migrationen gemacht werden.
Einführung	8	Durch die Migration beider Tables ist das Potential für Schwierigkeiten bei der Einführung hier am Grössten.
Wissen	3	Neues Wissen wird kaum erlangt, als Bonus gilt lediglich das Schreiben einer Migration.
Performance	7	Performance könnte durch erhöhte Aufrufe auf einen Table gefährdet sein.

Tabelle 12.8: Bewertung Lösungsvariante 3

		Variante 1		Variante 2		Variante 3	
Kriterium	Gewichtung	Ungewichtet	Gewichtet	Gewichtet	Gewichtet	Ungewichtet	Gewichtet
Zeitaufwand	40%	2	0.8	4	1.6	4	1.6
Einführung	15%	4	0.6	5	0.75	8	1.2
Wissen	30%	10	3	8	2.4	3	0.9
Performance	15%	5	0.75	5	0.75	7	1.05
Total	100%	21	5.15	19	5.5	22	4.75

Tabelle 12.9: Nutzwertanalyse

Fazit

Aus der Nutzwertanalyse kann geschlossen werden, dass sich die Variante 2 am Besten für die IPA eignet. Sie ist in der nötigen Zeit umsetzbar, bietet die Möglichkeit zum Wissensausbau, kann schnell eingeführt werden und wird vorausichtlich an der Performance nichts verändern.

12.2.5 Ausarbeitung

Im Diagramm unten ist die augearbeitete Klassenstruktur zu sehen. Alles was grün markiert ist, wurde neu hinzugefügt oder erweitert, alles was rot ist wurde entfernt. Es wird ein neues Modul geben, das `ModelFilter` Modul welches die Filterprozesse übernimmte welche vorher von der `Person::Filter::List` und `MailingLists::Subscribers` Klasse durchgeführt wurden.

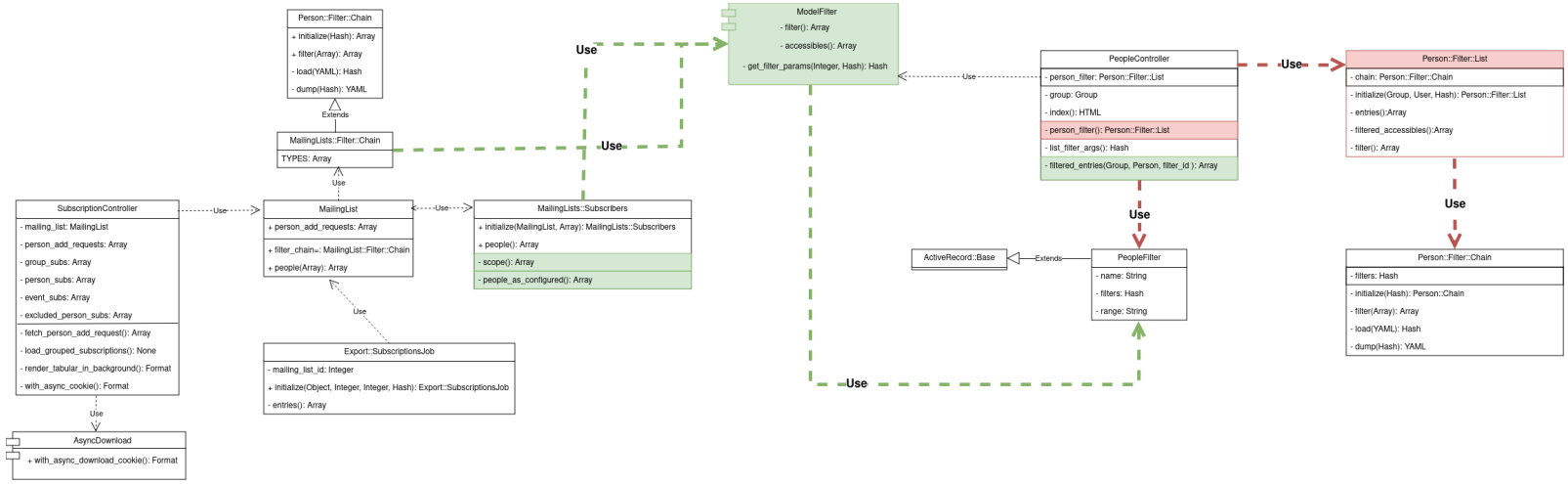


Abbildung 12.8: Ausgearbeitete Klassenstruktur

Um diese Klassen und Funktionen zu ersetzen, sind drei Methoden im neuen Modul angedacht. Zum einen die `filter` Methode welche auf die `Person Filter Chain` oder `MailingList Filter Chain` zugreifen wird, um den gegebenen Scope zu filtern. Damit nicht unberechtig auf Datensätze zugegriffen wird, gibt es zusätzlich die Methode `accessibles` welche die Daten vor der Fitlerung auf die Berechtigungen des Benutzers prüft. Zuletzt gibt es die Methode `get_filter_parameters` diese Methode ist dazu angedacht, um auf das `PeopleFilter` Model zuzugreifen und die Filterkriterien aus diesem herauszuholen, wenn ein solcher Filter existiert.

12.2.6 Gems

TODO: Describe

12.3 Sicherheitskonzept

Um die Sicherheit im Umbau der Filter sicherzustellen wird ein Sicherheitskonzept benötigt. Das Ziel dieses Konzeptes ist es mögliche Angriffe aufzuführen und die Blockade dieser Angriffe zu dokumentieren.

12.3.1 SQL-Injection

Die Einzigen Userinputs welche in dieser IPA auftreten sind die Textinputs welche im Filter ausgewählt werden können. Da die Eingabe des Users nicht direkt in die Postgres Datenbank gespeichert wird, ist diese Eingabe nicht für eine SQL-Injection gefährdet. Selbst wenn die SQL-Abfrage direkt gemacht würde, verhindert das ORM ActiveRecord mit seinen Standardmethoden, dass schädliche Eingaben abgespeichert werden. Dies geschieht unter anderem durch escapen der Strings.

12.3.2 Cross-Site Scripting

Da in diesem Feature Userinputs an das Rails-Backend gesendet werden muss die Cross-Site Scripting Attacke ebenfalls berücksichtig werden. Rails selbst bietet dafür einen eingebauten Abwehrmechanismus. Wenn der User eine XSS-Attacke ausführen möcht, indem er in einem der Filterinputs folgende Javascript Eingaben macht:

```
<h2>Welcome <script>alert("This is a XSS attack!")</script></h2>
```

Standardmässig escaped Rails diese Eingaben und ändert die special characters. So wird aus der Eingabe oben:

```
<h2>Welcome &lt;script&gt;alert\
(&quot;This is a XSS attack!&quot;)&lt;/script&gt;</h2>
```

12.3.3 URL Interpretation

Bei der URL-Interpretation fabriziert der Angreifer eine URL um damit auf die persönlichen Daten eines Benutzers zuzugreifen. Dabei kann der Angreifer versuchen die URL zu erraten. Dieser Angriff wird in Hitobito mit dem Gem `can-can-can` verhindert. Mit diesem Gem wird sichergestellt, das der Absender der Anfrage die nötigen Berechtigungen für das Einsehen der Informationen hat. Die Prüfung der Berechtigungen sieht wie folgt aus:

```
class Ability include CanCan::Ability

def define_root_abilities
 can :manage, :all
# root cannot change her email, because this is what makes her root.
 cannot :update_email, Person do |p|
   p.root?
 end
end
```

12.3.4 Kommunikation HTTP/S

Die Umgebungen auf der Integration und Produktion kommunizieren via HTTPS. Somit ist die verschlüsselte Kommunikation beim Transfer von produktiven Daten gesichert.

12.4 Fehlerbehandlungskonzept

Bei der Entwicklung und während der Laufzeit können stets Fehler oder nicht vorgesehene Probleme entstehen. Im folgenden Abschnitt wird dokumentiert, wie mit diesen Fällen umgegangen wird.

12.4.1 Nutzereingabe

Bei der Nutzereingabe des Users werden keine möglichen Exceptions erwartet. Der Benutzer kann im Filter, in der Suche nach einem bestimmten Text alles eingeben, ohne Einschränkungen. Mögliche Angriffe über Injections werden nach dem definierten Sicherheitskonzept abgehandelt.

12.4.2 Laufzeitfehler

Tritt in der Applikation ein Laufzeit Fehler auf wird dies sowohl in den Logs wie in der Sentry Umgebung von Hitobito aufgezeigt. Im Sentry werden zusätzlich die aufgetretenen Execptions gesammelt, um den Entwicklern eine Übersicht über allfällige Bugs zu geben. Gesammelte Exceptions können einem Entwickler zugewiesen oder wenn sie gefixed wurden, vom Sentry entfernt werden. Für diese IPA ist keine Modifizierung an der Sentry Umgebung nötig.

12.4.3 Exception Handling

Ruby benötigt kein Exception Handling wie es in anderen Sprachen der Fall ist. Exceptions werden durch Conditionals abgehandelt, reicht das nicht werden die Exceptions wie die Laufzeitfehler behandelt und erscheinen im Sentry.

12.5 Testkonzept

12.5.1 Testinfrastruktur

In dieser IPA unterscheiden wir drei Arten von Tests:

- **Controller Tests:** Tests die Action eines Controllers
- **Domain Tests:** Testet die Funktion eines Domain-Models
- **Manuelles Testen:** Testet das gesamte Feature

Hitobito verwendet für das Ausführen lokaler Tests RSpec 3.13.0. Für die manuellen Tests wird Firefox 80.0 (64-bit) verwendet. Als Gerät wird ein Laptop mit Pop!_OS 22.04 LTS verwendet. Um während der Entwicklung die Funktionalität der Applikation sicherzustellen werden die bereits bestehenden Tests im Hitobito ausgeführt und somit die Controller sowie Domain Tests abgedeckt. Es werden in dieser Arbeit somit keine neuen Domain oder Controller Tests geschrieben.

12.5.2 Fehlerklassen

Bezeichnung	Fehlerklasse	Beschreibung
FK0	Fehlerfrei	Keine Fehler
FK1	Nicht erfolgs-gefährdend	Kleine Fehler, beeinträchtigen Funktion nur bedingt.
FK2	Erfolgs-gefährdend	Fehler welche die Funktion beein-trächtigen.

Tabelle 12.10: Fehlerklassen

12.5.3 Manuelle Tests

13 Ausführung

13.1 Klassenstruktur

Während der Umsetzung habe ich mich an der Klassenstruktur im Entwurf orientiert. Obwohl die Grundlage der Implementation diesem Konzept gefolgt ist, gabe es dennoch gewisse Abweichungen. Entsprechend wurde ein neues Klassendiagramm erstellt:

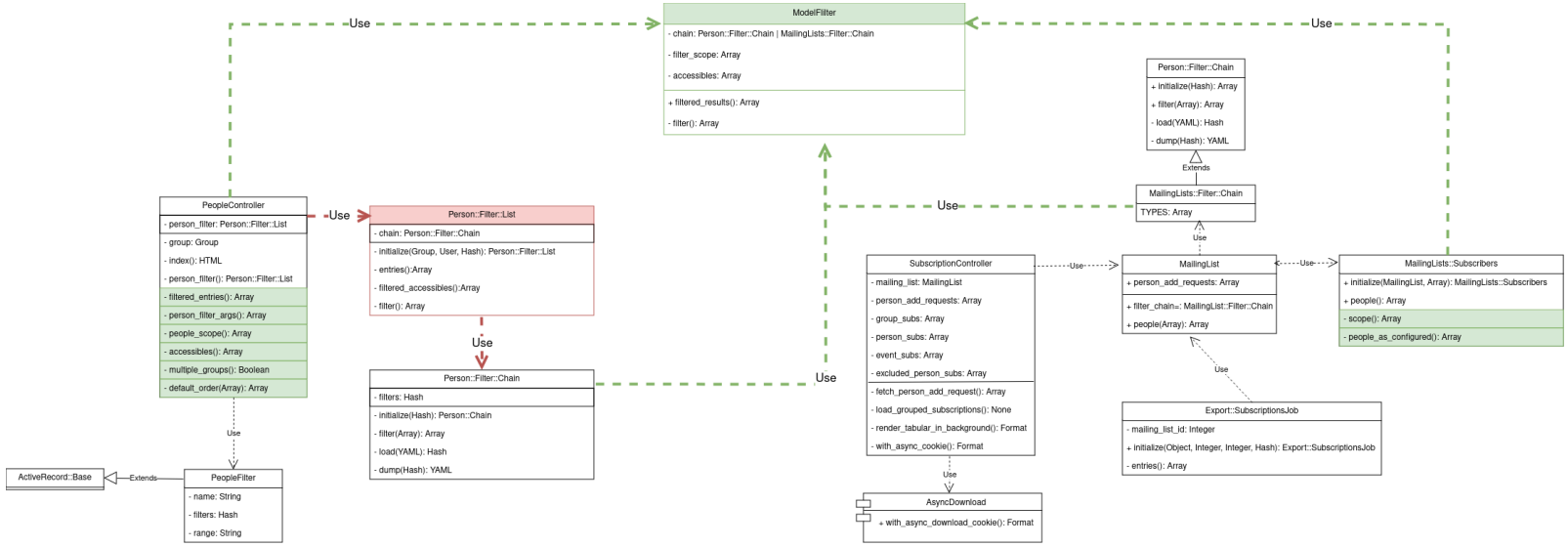


Abbildung 13.1: Klassendiagramm nach Umsetzung

13.1.1 PeopleController

Der PeopleController musste mehr erweitert werden als gedacht. Dies lag an der Entscheidung jegliche `People`Logik aus dem Filter zu entfernen und in den Controller zu verlagern. Die wichtigste Methode ist `filtered_entries`. Von dieser Methode aus wird der Aufruf zum `ModelFilter` gemacht, welcher dann die Filterung übernimmt. Aus dem Filter selbst wurde die Methode `default_order` übernommen, welche dafür sorgt, dass die Ergebnisse in der richtig sortiert werden. Vorher befand sich die Methode im Filter, sie wurde entfernt da die Sortierung nichts mit der Filterung zu tun hat und sich mit der neuen Struktur eine klare Abgrenzung zwischen Filterungssystem und Formatierung der Daten vor dem Zurückstellen ergibt.

13.1.2 ModelFilter

Der `ModelFilter` kam in der Grössenordnung in der er im Entwurf geplant wurde daher. Anders als geplant wurde er als Klasse nicht als Modul implementiert. Der Grund dafür ist, das in einer View eine Instanz des Filter benötigt ist, um auf nur ihm bekannte Attribute zuzugreifen. Als Zusatz kam das `chain`-Argument. Während des Entwickelns kam die Idee auf, dass anstatt einer get-Methode für die Filterchain, diese direkt vom Controller und der Subscriber-Klasse übergeben werden könnte. So wird das geplante `type`-Attribut beim Filteraufruf gespart und der Filter wird noch generischer. Dies führt dazu das die nicht funktionale Anforderung 1 (Erweiterbarkeit) besser abgedeckt werden kann, denn nun könnten theoretisch jegliche weitere Models mit ihrer eigenen Filterchain an den `ModelFilter` gegeben werden, welcher nur noch die Filterung selbst durchführt.

13.1.3 MaillingLists::Subscribers

Diese Klasse kam dem Entwurf am nächsten. Es konnte wie geplant die Methoden `scope` und `people_as_configured` so erweitert werden, dass nur noch der Aufruf zum `ModelFilter` stattfindet. Die restliche Logik bleibt unverändert.

13.1.4 Person::Filter::List

Im UML wurde diese Klasse mit rot markiert, da sie nun nicht mehr im Filterprozess der Personenlisten und Abonnementen integriert ist. Um die Funktionalität der Komponenten, welche in dieser IPA nicht berücksichtig trotzdem zu gewährleisten und somit die Möglichkeit für manuelles Testing sicherzustellen, wurde die Klasse nicht aus dem Repository entfernt.

13.2 Gems

In der Analyse der Arbeit wurde aufgeführt welche Gems voraussichtlich in der Arbeit verwendet werden. Dieser Abschnitt klärt, wie die Arbeit mit den genannten Gems funktionierte.

13.2.1 can-can-can

In der Analyse wurde erwartet das mit dem can-can-can Gem gearbeitet werden muss, um nur die Daten in die Filterung aufzunehmen, auf welche der Benutzer Zugriff hat. Wie während der Implementierung festgestellt wurde, konnte direkt auf ein Array mit den bereits verifizierten Daten zugegriffen werden. Sodurch war eine direkte Interaktion mit dem Gem nicht mehr nötig.

13.2.2 dry-crud

Das dry-crud Gem wurde ebenfalls nicht direkt verwendet. Die Anpassungen am Controller konnten vollzogen werden, ohne Methoden zu verändern welche durch von diesem Gem betroffen sind.

13.3 Unvorhergesehene Änderungen

In der Umsetzung dieser Arbeit traten Änderungen an Teilen der Applikation auf, welche im Entwurf nicht berücksichtig wurden.

13.3.1 application.rb

Standardmässig inkludiert Rails nur bestimmte Ordner und Dateien in seinem `load_path`. Während der Implementation wurde ein neuer Ordner `filter` angelegt um die neue Klasse darin zu verwalten. Da diese Klasse nicht im `load_path` war musste sie im `application.rb` hinzugefügt werden:

```
# Custom directories with classes and modules you want to be autoloadable.
 config.autoload_paths += %W( #{config.root}/app/abilities
                              #{config.root}/app/domain
                              #{config.root}/app/domain/filter
                              #{config.root}/app/jobs
                              #{config.root}/app/serializers
                              #{config.root}/app/utils
                            )
```

13.3.2 _list.html.haml

Die View welche die Personenliste darstellt, beinhaltete eine Referenz adressiert an die Instanz der `Person::Filter::List`. Damit die View dennoch für das manuelle Testing verwendete werden kann wurde ein Teil der Zeit für die Sicherstellung der Anzeige verwendet. Der Teil der am meisten Aufwand verursachte war die Anpassung des `FilterNavigation::People` in der View.

```
- content_for(:filter, FilterNavigation::People
.new(self, @group, @model_filter, @name).to_s)
```

Dieser Komponent verwaltet die Auswahl des Filers über das UI. Durch das Entfernen der `Person::Filter::List` aus dem Filterprozess, musste die Refernz der View auf den neuen `ModelFilter` geändert werden. Zusätzlich holte dieser Komponent den definierten Name des Personenfilters:

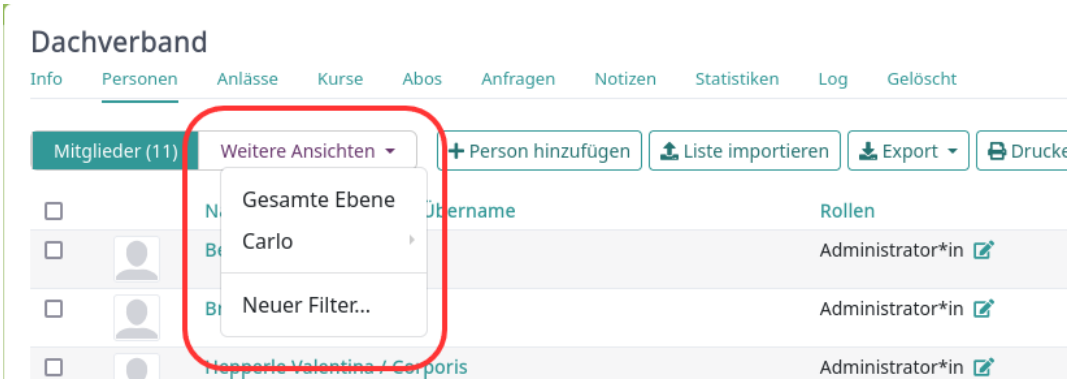


Abbildung 13.2: Klassendiagramm nach Umsetzung

Über diese Referenz. Das das Ziel eine Abgrenzung genau diser Verbindung der Funktionen war, wurde eine Instanzvariable im `PeopleController` angelegt welche den Namen des Personenlistenfilters speichert falls dieser existiert:

```
  def person_filter_args
   if params[:filter_id]
   filter = PeopleFilter.for_group(group).find(params[:filter_id])
     @name = filter.name
     filter.to_params[:filters]
   else
     params
   end
end
```

13.4 Testprotokoll

Resultat Testfall Nr. 1	
Testname	
Testkontext	
Testperson	
Ausführungs Datum	
Testergebnis	
Beschreibung	
Fehlerklasse	

Tabelle 13.1: Resultat Testfall 1

14 Einführung

15 Sprintabschlüsse

15.1 Abschluss Sprint Initialisierung

15.2 Abschluss Sprint Umsetzung

15.3 Abschluss Sprint Finalisierung

Teil III

Anhänge und Verzeichnise

Hitobito: Neue Generation von Personen-Filtern
Autor: Marc Egli

16 Verzeichnise

16.1 Tabellenverzeichnis

16.2 Abbildungsverzeichnis

16.3 Code Verzeichnis

Quellenverzeichnis

[TODO: Name der Quelle] TODO:URLeinf\protect\unhbox\voidb@x\bgroup\U@D1ex{\setbox\z@\hbox{\char127}\dimen@-.45ex\advance\dimen@\ht\z@}\accent127\fontdimen5\font\U@Du\egroupgen, (TODO: Datum von Tag wo Quelle verwendet wurde)

17 Verwendete Abkürzungen

Abkürzung	Bedeutung
TODO: Abkürzung	TODO: Beschreibung

Tabelle 17.1: Verwendete Abkürzungen

18 Glossar

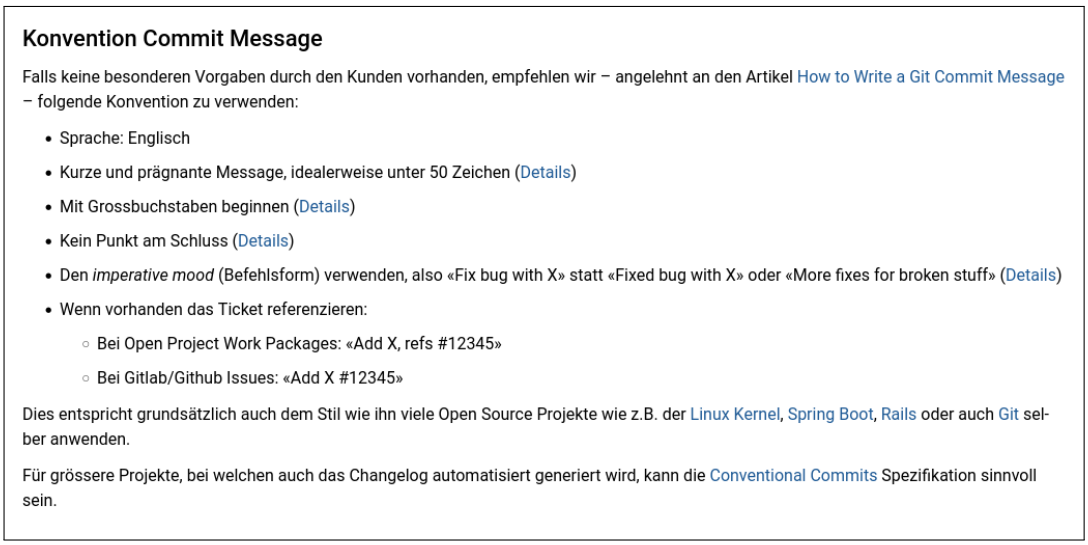
Bezeichnung	Bedeutung
TODO: Wort	TODO: Beschreibung

Tabelle 18.1: Glossar

19 Anhänge

19.1 Sitzungsprotokolle

19.2 Git commit convention



Konvention Commit Message

Falls keine besonderen Vorgaben durch den Kunden vorhanden, empfehlen wir – angelehnt an den Artikel How to Write a Git Commit Message – folgende Konvention zu verwenden:

- Sprache: Englisch
- Kurze und prägnante Message, idealerweise unter 50 Zeichen (Details)
- Mit Grossbuchstaben beginnen (Details)
- Kein Punkt am Schluss (Details)
- Den *imperative mood* (Befehlsform) verwenden, also «Fix bug with X» statt «Fixed bug with X» oder «More fixes for broken stuff» (Details)
- Wenn vorhanden das Ticket referenzieren:
  - Bei Open Project Work Packages: «Add X, refs #12345»
  - Bei Gitlab/Github Issues: «Add X #12345»

Dies entspricht grundsätzlich auch dem Stil wie ihn viele Open Source Projekte wie z.B. der Linux Kernel, Spring Boot, Rails oder auch Git selber anwenden.

Für grössere Projekte, bei welchen auch das Changelog automatisiert generiert wird, kann die Conventional Commits Spezifikation sinnvoll sein.

Abbildung 19.1: Puzzle ITC Git commit conventions

19.3 Security conventions

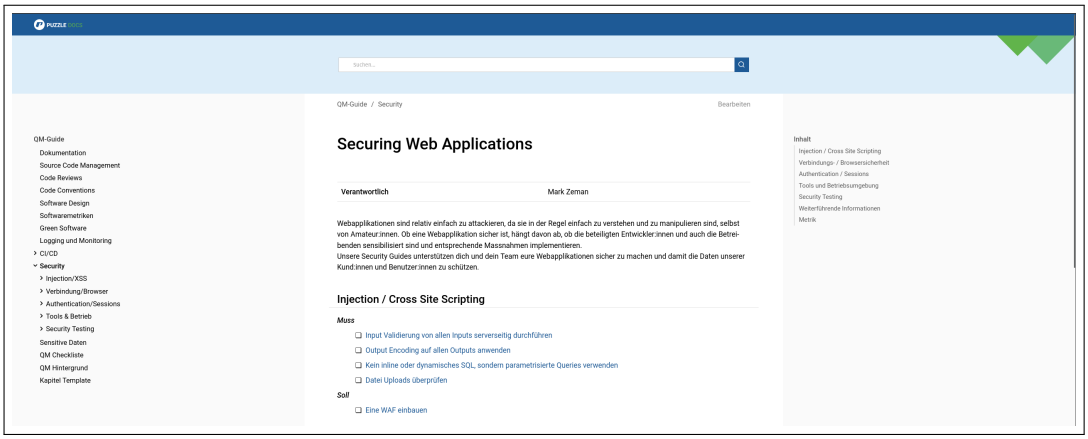


Abbildung 19.2: Puzzle ITC security conventions 1/3

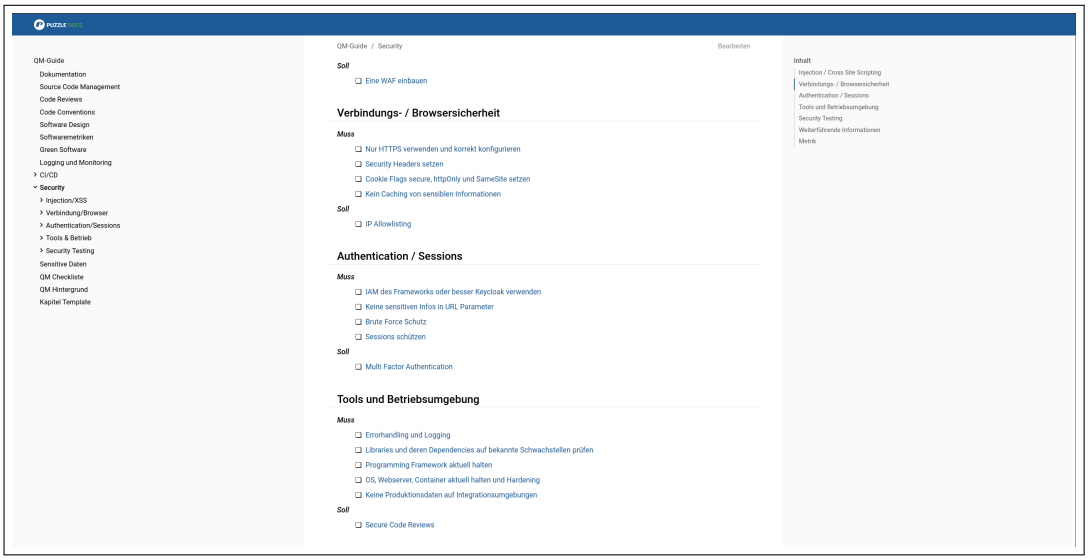


Abbildung 19.3: Puzzle ITC security conventions 2/3

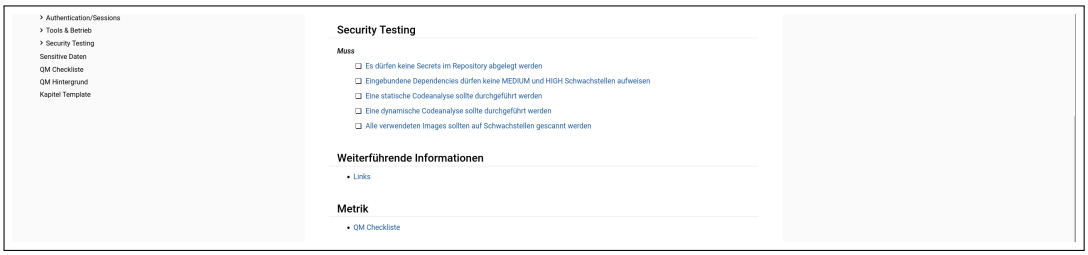


Abbildung 19.4: Puzzle ITC security conventions 3/3

19.4 Datenschutzkonzept