

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	

Tabelle 1: IPA Daten

Teil I

Ablauf, Organisation und Umfeld

Inhaltsverzeichnis

I	Ablauf, Organisation und Umfeld	1
1	Aufgabenstellung	6
1.1	Titel der Arbeit	6
1.2	Ausgangslage	6
1.3	Detaillierte Aufgabenstellung	7
1.3.1	Mittel und Methoden	9
1.3.2	Vorkenntnisse	10
1.3.3	Vorarbeiten	10
1.3.4	Neue Lerninhalte	10
1.3.5	Arbeiten in den letzten 6 Monaten	10
2	Firmenstandards	11
2.1	Code conventions	11
2.1.1	Mehrsprachigkeit	11
2.1.2	Lizenz	11
2.2	Git Commit Message Conventions	12
3	IPA-Schutzbedarfsanalyse	13
3.1	Datensicherheit	13
3.2	Applikationssicherheit	13
4	Organisation der IPA-Ergebnisse	14
4.1	Datensicherung	14
4.1.1	Dokumentation	14
4.1.2	Code	14
4.1.3	Wiederherstellung des Codes	14
5	Projektmethode	15
5.1	Einsatz von Scrum	15
5.1.1	Sprints	15
5.1.2	Verwaltungstool	15
5.1.3	Meetings	16
5.1.4	Abweichungen	17
5.2	Definition of Done	18
5.2.1	DoD Code	18
5.2.2	DoD Dokumentation	18
5.2.3	Akzeptanzkriterien	19

5.3	Verwendungsgrund	19
6	Projektaufbauorganisation	20
6.1	Projektrollen in Scrum	20
6.2	Projektrollen IPA	20
6.3	Rollenverteilung	20
7	Zeitplan	21
7.1	Erläuterung zum Zeitplan	21
7.2	Sprints	21
8	Arbeitsjournale	22
8.1	Tag 1: Datum	22
8.2	Tag 2: Datum	23
8.3	Tag 3: Datum	24
8.4	Tag 4: Datum	25
8.5	Tag 5: Datum	26
8.6	Tag 6: Datum	27
8.7	Tag 7: Datum	28
8.8	Tag 8: Datum	29
8.9	Tag 9: Datum	30
8.10	Tag 10: Datum	31
8.11	Tag 11: Datum	32
8.12	Tag 12: Datum	33
9	Persönliches Fazit	34
9.1	Was lief weniger gut	34
9.2	Was lief gut	34
9.3	Schlussreflexion	34
II	Projektdokumentation	35
10	Einführung	36
11	Analyse	37
11.1	Ist-Zustand	38
11.1.1	Personenlisten	38
11.1.2	Abonnemente	38
11.2	Soll-Zustand	38
11.3	Bedürfniserhebung	38
11.4	Risikoanalyse	38
11.5	Anforderungen	38
11.5.1	Nicht funktionale Anforderungen	38
11.5.2	Funktionale Anforderungen	38
11.6	Abgrenzung	38
11.7	Benötigter Rahmen	38

11.7.1 Fehlende Informationen	38
11.8 Persönliche Vorgehensziele	38
12 Entwurf	39
12.1 Anwendungskonzept	40
12.1.1 Anwendungsdiagramm	40
12.1.2 Anwendungsfälle	40
12.2 Systemkonzept	40
12.2.1 Betroffene Services	40
12.2.2 Status quo	40
12.2.3 Lösungsvarianten	40
12.2.4 Variantenentscheid	40
12.3 Sicherheitskonzept	40
12.3.1 SQL-Injection	40
12.3.2 Cross-Site Scripting	40
12.3.3 URL Interpretation	40
12.3.4 Kommunikation HTTP/S	40
12.4 Fehlerbehandlungskonzept	40
12.4.1 Nutzereingabe	40
12.4.2 Laufzeitfehler	40
12.5 Testsetup	40
12.6 Testkonzept	40
12.6.1 Testinfrastruktur	40
12.6.2 Fehlerklassen	40
12.6.3 Manuelle Tests	40
13 Ausführung	41
13.1 Einsatz von KI-Modellen	41
13.2 Gems	41
13.2.1 can-can-can	41
13.2.2 dry-crud	41
14 Einführung	42
14.1 Instruktion	42
14.2 Unvorhergesehene Änderungen	42
14.2.1 application.rb	42
14.2.2 _list.html.haml	42
15 Sprintabschlüsse	43
15.1 Abschluss Sprint Initialisierung	43
15.1.1 Backlog	43
15.2 Abschluss Sprint Umsetzung	43
15.2.1 Backlog	43
15.3 Abschluss Sprint Finalisierung	43
15.3.1 Backlog	43

III	Anhang und Verzeichnisse	44
16	Verzeichnisse	45
16.1	Code	45
16.2	Tabellenverzeichnis	45
16.3	Abbildungsverzeichnis	45
	Quellenverzeichnis	46
17	Verwendete Abkürzungen	47
18	Glossar	48
19	Anhänge	49
19.1	Git Commit Message Convention	49
19.2	Daily-Protokolle	49
19.3	Sitzungsprotokolle	49
19.4	Git commit convention	49
19.5	Security conventions	49

1 Aufgabenstellung

1.1 Titel der Arbeit

Hitobito: Neue Generation von Personen-Filtern

1.2 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: [Hitobito](#)

Eine Kernfunktionalität von Hitobito ist das Filtern von Personenlisten und von Mailinglistenempfängern mit konfigurierbaren Filtern. Diese werden über das Webinterface konfiguriert. Das Webinterface wurde mit statischen Webtechnologien entwickelt und ist inzwischen ziemlich in die Jahre gekommen.

Eine Erneuerung dieser Komponente ist ein Wunsch vieler Kunden.

1.3 Detaillierte Aufgabenstellung

Mit dieser IPA soll ein neues UI mit Hotwire für die Persistierung von Filter-Parametern im Hitobito Generic-Wagon erstellt werden (rein Frontend).

- Die Ansichten zur Konfiguration für Filter der Personenlisten und Abonnemente werden mit dem neuen UI ersetzt.
- Die neuen Ansichten werden nach einem gegebenen Mockup umgesetzt. Dieses Mockup wurde vom Kandidaten in Zusammenarbeit mit einem UX Experten erarbeitet und muss als Grundlage für die Ausarbeitung des Interfaces verwendet werden. Des weiteren muss das Interface auf das visuelle Design der existierenden Applikation abgestimmt sein.
- Das Backend darf nicht angepasst werden, das heisst das neue Interface verwendet die bestehenden Endpunkte und schickt die Daten im selben Format wie das alte Interface. Dies muss mit automatisierten Tests sichergestellt werden.
- Formular zur Konfiguration von Personen-Listen Filter: Das bestehende Formular muss ersetzt werden durch eine neue Implementation mit den in Mittel und Methoden definierten Web Technologien. Diese neue Umsetzung muss es erlauben, dynamisch weitere Filterkriterien hinzuzufügen im Gegensatz zur alten Implementation welche mit einem statischen Formular implementiert ist.
- Formular zur Konfiguration von Abo-Empfänger Filter: Das bestehende Formular besteht aus mehreren Teilen, wovon im Rahmen der IPA nur der Teil für die Globalen Filterbedingungen angepasst werden muss. Wie bei den Personen-Listen Filter muss das Formular nun dynamisch implementiert werden. Die Formulare für die weiteren Filterbedingungen werden im Rahmen der IPA nicht angepasst.
- Code der während dieser IPA entsteht soll auf ein privates Github Repo gepushed werden. Die VFs haben dabei stets Lese-Rechte.
- Die Konventionen des Ruby Style Guide, des Rails Style Guide und für Git Commit Messages müssen eingehalten werden (siehe Mittel und Methoden).

Out of Scope - wird erst nach der IPA umgesetzt:

- Filterung für Rollen, Gruppen, Events, People bei Abonnements.
- Anpassungen der Ansicht in den anderen Wagons.
- Anpassungen der bisher bestehenden Tests in Hitobito welche die zu erweiternden Ansichten betreffen.

Weitere Anforderungen zu spezifischen Bewertungskriterien:

- G1: Dokumentation fachlicher und technischer Anforderungen: Die fachlichen und technischen Anforderungen müssen dokumentiert werden.
- G10: Konforme Implementierung und Versionierung: Applikationen und Schnittstellen müssen konform implementiert und versioniert werden.
- A13: Erhebung und Dokumentation der Bedürfnisse und Umfeld: Die Bedürfnisse und das Umfeld werden adäquat erhoben und dokumentiert.
- A15: Instruktion: Es wird für den Projektowner eine Instruktion durchgeführt. Diese muss dem Projektowner die relevanten Änderungen aufzeigen.
- C11: Einsatz von KI-Modellen: Wir setzen bei Puzzle KI in Form von Kopiloten und Chatbots als Hilfsmittel ein. Die Lernenden werden im sinnvollen Einsatz von solcher KI geschult. Dies umfasst z.B. den Umgang in Bezug auf Output Validierung, Transparenz und Sicherheit. Die IPA soll möglichst repräsentativ für unseren Alltag als Entwickler sein, dementsprechen darf KI ein Teil davon sein.
- G5: Risikoanalyse und Sicherheitsmassnahme: Sicherheitsrisiken von Applikationen und Schnittstellen müssen identifiziert und adressiert werden.
- G6: Entwicklung und Anpassung des Anforderungskatalogs: Ein Anforderungskatalog für Sicherheitsmassnahmen von Applikationen und/oder Schnittstellen muss erstellt oder angepasst werden.
- User Experience und visuelles Design: Das Feature muss visuell gut gestaltet sein um die Usability und Nutzerfreundlichkeit des Features sicherzustellen.

- Versionsverwaltung mit Git (Source Code): Die Versionsverwaltung mit Git muss gemäss den Best Practices erfolgen. Es müssen sprechende und einheitliche Commit-Messages geschrieben werden und commit-spezifische Inhalte müssen passend zur Message sein und unter der Einhaltung der Firmenguidelines erfolgen.
- Bewertung von Aussagen: Aussagen in der Arbeit müssen klar zwischen persönlichen Meinungen und auf Quellen basierenden Informationen differenziert werden.

1.3.1 Mittel und Methoden

Technologie und Plattform:

- Ruby, Ruby on Rails, Active Record
- HTML, CSS, Javascript, Hotwire
- PostgreSQL
- Git

Entwicklungsumgebungen:

- IntelliJ
- Visual Studio Code
- Github
- Rake
- Rubocop

Textverarbeitung und Diagramme

- Latex
- draw.io

1.3.2 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.

1.3.3 Vorarbeiten

- Vorbereitung Dokumentvorlage
- Probe-IPA: Vereinheitlichung der Personenlisten- und Abonnementsfilterlogik im Backend
- Entwurf eines Mockups

1.3.4 Neue Lerninhalte

- Eigenständiges Umsetzen eines Designs nach gegebenem Mockup
- Eigenständiges Projektmanagement während der IPA

1.3.5 Arbeiten in den letzten 6 Monaten

- Umsetzung diverser Features und Bugfixes für Hitobito (Ruby on Rails)
- Probe-IPA: Vereinheitlichung der Personenlisten- und Abonnementsfilterlogik
- PostgreSQL Migration Hitobito
- Ruby on Rails Major Upgrade 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 Mehrsprachigkeit

Hitobito ist eine mehrsprachige Applikation. Alle Erweiterungen oder Anpassungen müssen in Deutsch übersetzt werden. Übersetzungen werden in einer Übersetzungsdatei gespeichert oder können vom Kunden in einem Tool namens Transifex verwaltet werden.

2.1.2 Lizenz

Hitobito ist ein Open Source Projekt. In jedem File in Hitobito wird das Copyright für den jeweiligen Kunden in Kommentarform beschrieben. Diese Lizenz- und Kundeninformationen können über folgenden Befehl eingefügt werden:

```
rake license:insert
```

Die daraus entstehende Lizenz sieht wie folgt aus:

```
1  # Copyright (c) 2012 -2021 , hitobito AG . This file is part of
2  # hitobito and licensed under the Affero General Public License version 3
3  # or later . See the COPYING file at the top - level directory or at
4  # https :// github . com / hitobito / hitobito .
```

Alternativ dazu können diese Informationen mit

```
rake license:remove
```

entfernt oder mit

```
rake license:update
```

aktualisiert werden.

2.2 Git Commit Message Conventions

Die Git Commit Messages werden nach den Regeln von Puzzle ITC formuliert. Im Anhang unter Git Commit Message Conention finden sie eine Kopie der Firmenkonventionen. Diese wurden basierend auf folgendem Tutorial definiert: [Tutorial](#)

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

3 IPA-Schutzbedarfanalyse

3.1 Datensicherheit

3.2 Applikationssicherheit

4 Organisation der IPA-Ergebnisse

4.1 Datensicherung

4.1.1 Dokumentation

4.1.2 Code

4.1.3 Wiederherstellung des Codes

5 Projektmethode

Die verwendete Projektmethode dieser IPA ist Scrum. Im folgenden Abschnitt wird der Einsatz, Abweichungen, Werkzeuge und Begründung der Wahl dieser Projektmethode beschrieben. Des weiteren beschreibt dieser Abschnitt die Definition of Done (DoD).

5.1 Einsatz von Scrum

5.1.1 Sprints

Die IPA wird insgesamt in drei Sprints unterteilt. Jedem Sprint wird eine Phase der Arbeit zugewiesen. Die Aufteilung ist wie folgt:

- Sprint 1: Initialisierung
- Sprint 2: Umsetzung
- Sprint 3: Finalisierung

5.1.2 Verwaltungstool

Als Verwaltungstool wird Github Projects eingesetzt. Das Board hierzu kann unter [Github Board](#) aufgerufen werden. Das Board ist in sechs Spalten unterteilt:

- Backlog: User-Stories werden grob erfasst, keine Details nötig.
- Refinement: User-Stories werden genauer Beschrieben und Akzeptanzkriterien werden definiert.
- Ready: User-Story wurde refined und geschätzt. Sie kann jetzt bearbeitet werden.
- In-Progress: User-Story wird momentan bearbeitet.

- In-Review: User-Story wurde abgeschlossen, alle Akzeptanzkriterien sind erfüllt.
- Done: User-Story erfüllt DoD (Definition of Done).

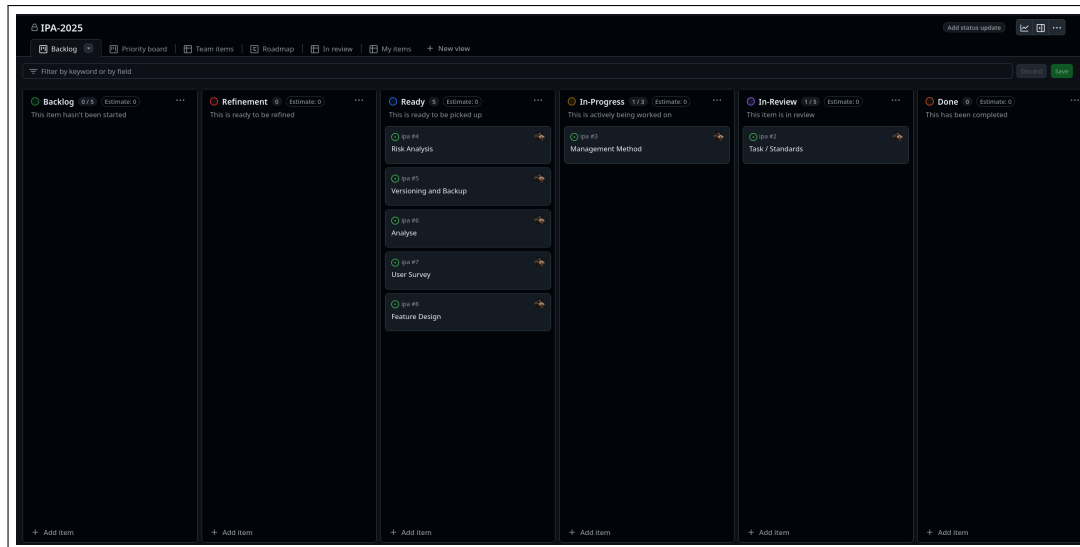


Abbildung 5.1: Github Projects Board

5.1.3 Meetings

Sprint Planning

Zu Beginn eines Sprints werden werden alle Aufgaben in Form von User-Stories im Backlog erfasst. Die Stories werden anschliessend refined und danach geschätzt. Das Sprint Planning umfasst den Prozess der Erfassung von User-Stories, deren Refinement und Schätzung. Konnten im letzten Sprint die geplanten User-Stories nicht alle abgeschlossen werden, umfasst das Planning zusätzlich das Neurefinement und die Neuschätzung dieser User-Stories. Anwesend beim Sprint Planning ist ausschliesslich der Kandidat.

Dailies

Während eines Sprints wird jeden Tag um 09:00 Uhr ein Daily durchgeführt. Das Daily findet bei Puzzle ITC im Sitzungszimmer SSudoßtatt. Anwesend sind dabei der Kandidat, die verantwortliche Fachkraft und die zusätzliche verantwortliche Fachkraft. Ausgenommen von dieser Regel ist der erste Tag der IPA (04.03.2025) an welchem kein Daily durchgeführt wird. Grund dafür ist, dass zu diesem Zeitpunkt noch keine Organisation und

Projektvorgehensweise definiert wurde und die ersten Prozesse von Scrum erst ab dem 2. Tag eintreffen können.

Im Daily ist es dem Kandidat möglich, Fragen an seine verantwortlichen Fachkräfte zu stellen. Jedes Daily wird protokolliert. Die Protokolle der Dailies können unter Daily-Protokolle eingesehen werden.

Sprintabschlüsse

Nach jedem Sprint findet ein einstündiges Meeting für den Sprintabschluss statt. Darin werden die abgeschlossenen User-Stories in der In-Review-Spalte verifiziert. Erfüllt die hinterlegte Arbeit alle Akzeptanzkriterien wird die User-Story auf Done geschoben. Sind die Akzeptanzkriterien nicht erfüllt, wird die User-Story auf Refinement geschoben. Anwesend beim Sprintabschluss ist ausschliesslich der Kandidat. In Folge des Sprintabschlusses wird das Sprint Planning durchgeführt.

5.1.4 Abweichungen

Trotz der Verwendung von Scrum, wurden Änderungen an der Definition dieser Projektvorgehensmethode vorgenommen. Grund dafür ist, dass Scrum durch die Änderungen besser auf die IPA zugeschnitten ist.

Schätzung

Scrum verzichtet auf Schätzungen in Personenstunden und verwendet deswegen eine Währung namens „Story Points“. Story Points werden der [Fibonacci-Zahlenreihe](#) folgend vergeben. Der Sinn dabei ist, der Schätzung einer User-Story nach Personenstunden auszuweichen.

Dieses Konzept wird in dieser IPA verworfen, um in der Lage zu sein einen Zeitplan mit genauen Angaben in Personenstunden zu erstellen. Dies macht es dem Kandidaten möglich besser einzuschätzen, wie gut er in der Zeit liegt.

Abnahme Akzeptanzkriterien

Nach Scrum werden User-Stories vom Product Owner abgenommen. Um ständige Meetings mit dem Product Owner von Hitobito und den mithergehenden Zeitverlust zu vermeiden, werden die User-Stories vom Kandidaten selbst abgenommen. Den Prozess dazu finden ist unter Sprintabschlüsse ersichtlich.

Sprint Retro

Das Sprint Retro bietet dem Product Owner eine Möglichkeit einen Überblick über die Stimmung im Entwicklerteam zu erhalten. Sprint Retros finden im Geschäftsalltag Monatsweise statt. Auf das Sprint-Retro wird in dieser Arbeit verzichtet. Grund ist der kleine Zeitrahmen der IPA, welcher es unnötig macht ein solches Meeting durchzuführen.

5.2 Definition of Done

Die Definition of Done definiert wann eine User-Story abgeschlossen werden kann. Eine User-Story kann erst abgeschlossen werden, wenn sie alle Kriterien der Definition of Done erfüllt. Im Rahmen der IPA werden zwei Definition of Done's verwendet. Eine für User-Stories welche den Code betreffen, eine zweite für User-Stories welche die Dokumentation betreffen.

5.2.1 DoD Code

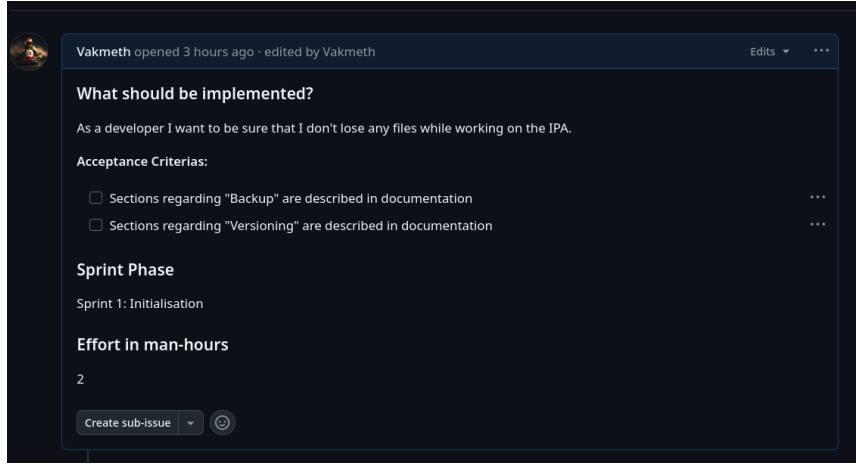
- Nur notwendige Konsolenausgaben vorhanden
- Feature relevante Tests vorhanden
- Sprechender Code implementiert
- Nicht verwendete Methoden gelöscht
- Feature manuell getestet
- Alle Akzeptanzkriterien erfüllt

5.2.2 DoD Dokumentation

- Definierte Sektion beschreiben
- Kriterien aus Kriterienkatalog erfüllt
- Kriterien gemäss Dokumentenvorlage erfüllt
- Keine Grammatik- / Rechtschreibfehler vorhanden
- Quellen angegeben

5.2.3 Akzeptanzkriterien

Die Akzeptanzkriterien einer User-Story werden im dazugehörigen Ticket verwaltet. Jede User-Story wurde nach einem definierten Template erstellt, welches in Github hinterlegt wurde. Eine User-Story kann folgendermassen aufgebaut sein:



The screenshot shows a GitHub issue template for a User Story. The template is titled 'What should be implemented?' and includes the following sections:

- What should be implemented?**
As a developer I want to be sure that I don't lose any files while working on the IPA.
- Acceptance Criteria:**
 - ☐ Sections regarding "Backup" are described in documentation
 - ☐ Sections regarding "Versioning" are described in documentation
- Sprint Phase**
Sprint 1: Initialisation
- Effort in man-hours**
2

At the bottom, there is a 'Create sub-issue' button and a 'Create' button.

Abbildung 5.2: Example of User Story

5.3 Verwendungsgrund

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

- **Sprint Ende:** SCRUM zwingt den Entwickler dazu am Ende des Sprints ein vorzeigbares 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

6.2 Projektrollen IPA

6.3 Rollenverteilung

7 Zeitplan

7.1 Erläuterung zum Zeitplan

7.2 Sprints

8 Arbeitsjournale

8.1 Tag 1: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.1: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.2 Tag 2: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.2: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.3 Tag 3: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.3: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.4 Tag 4: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.4: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.5 Tag 5: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.5: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.6 Tag 6: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.6: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.7 Tag 7: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.7: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.8 Tag 8: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.8: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.9 Tag 9: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamtstunden soll	Gesamtstunden ist

Tabelle 8.9: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.10 Tag 10: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.10: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.11 Tag 11: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.11: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

8.12 Tag 12: Datum

Tätigkeiten	Beteiligte Personen	Aufwand Geplant (std)	Aufwand Effektiv (std)
Tätigkeit	Personen	Stunden soll	Stunden ist
Total		Gesamt-stunden soll	Gesamt-stunden ist

Tabelle 8.12: Tätigkeiten Tag 1

Tagesablauf

Hilfestellungen

- Person: Hilfestellung

Reflexion

Was lief gut

Was lief weniger gut

Meine Erkenntnisse von heute

Nächste Schritte

9 Persönliches Fazit

9.1 Was lief weniger gut

9.2 Was lief gut

9.3 Schlussreflexion

Teil II

Projektdokumentation

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

10 Einführung

11 Analyse

11.1 Ist-Zustand

11.1.1 Personenlisten

11.1.2 Abonnemente

11.2 Soll-Zustand

11.3 Bedürfniserhebung

11.4 Risikoanalyse

11.5 Anforderungen

11.5.1 Nicht funktionale Anforderungen

11.5.2 Funktionale Anforderungen

11.6 Abgrenzung

11.7 Benötigter Rahmen

11.7.1 Fehlende Informationen

11.8 Persönliche Vorgehensziele

12 Entwurf

12.1 Anwendungskonzept

12.1.1 Anwendungsdiagramm

12.1.2 Anwendungsfälle

12.2 Systemkonzept

12.2.1 Betroffene Services

12.2.2 Status quo

12.2.3 Lösungsvarianten

12.2.4 Variantenentscheid

12.3 Sicherheitskonzept

12.3.1 SQL-Injection

12.3.2 Cross-Site Scripting

12.3.3 URL Interpretation

12.3.4 Kommunikation HTTP/S

12.4 Fehlerbehandlungskonzept

Version 1.0

4. März 2025

Seite 40 von 49

12.4.1 Nutzereingabe

12.4.2 Laufzeitfehler

13 Ausführung

13.1 Einsatz von KI-Modellen

13.2 Gems

13.2.1 can-can-can

13.2.2 dry-crud

14 Einführung

14.1 Instruktion

14.2 Unvorhergesehene Änderungen

14.2.1 application.rb

14.2.2 _list.html.haml

15 Sprintabschlüsse

15.1 Abschluss Sprint Initialisierung

15.1.1 Backlog

15.2 Abschluss Sprint Umsetzung

15.2.1 Backlog

15.3 Abschluss Sprint Finalisierung

15.3.1 Backlog

Teil III

Anhänge und Verzeichnisse

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

16 Verzeichnisse

16.1 Code

16.2 Tabellenverzeichnis

1	IPA Daten	1
8.1	Tätigkeiten Tag 1	22
8.2	Tätigkeiten Tag 1	23
8.3	Tätigkeiten Tag 1	24
8.4	Tätigkeiten Tag 1	25
8.5	Tätigkeiten Tag 1	26
8.6	Tätigkeiten Tag 1	27
8.7	Tätigkeiten Tag 1	28
8.8	Tätigkeiten Tag 1	29
8.9	Tätigkeiten Tag 1	30
8.10	Tätigkeiten Tag 1	31
8.11	Tätigkeiten Tag 1	32
8.12	Tätigkeiten Tag 1	33
17.1	Verwendete Abkürzungen	47
18.1	Glossar	48

16.3 Abbildungsverzeichnis

5.1	Github Projects Board	16
5.2	Example of User Story	19
19.1	Puzzle ITC Git commit conventions	49

Quellenverzeichnis

[Github Docs - Understanding connections between repositories]

[https://docs.github.com/en/repositories/
viewing-activity-and-data-for-your-repository/
understanding-connections-between-repositories](https://docs.github.com/en/repositories/viewing-activity-and-data-for-your-repository/understanding-connections-between-repositories), (04.03.2025)

[Github Docs - Configuring issue templates] [https://docs.github.com/en/
communities/using-templates-to-encourage-useful-issues-and-pull-requests/
configuring-issue-templates-for-your-repository](https://docs.github.com/en/communities/using-templates-to-encourage-useful-issues-and-pull-requests/configuring-issue-templates-for-your-repository), (04.03.2025)

[Leo - Translating] <https://dict.leo.org/german-english>, (04.03.2025)

17 Verwendete Abkürzungen

Abkürzung	Bedeutung
UML	Unified Modeling Language

Tabelle 17.1: Verwendete Abkürzungen

18 Glossar

Bezeichnung	Bedeutung
Hitobito	Community Management Tool

Tabelle 18.1: Glossar

19 Anhänge

19.1 Git Commit Message 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

19.2 Daily-Protokolle

19.3 Sitzungsprotokolle

19.4 Git commit convention

19.5 Security conventions